



Dr.WEB

for UNIX Internet Gateways

Administrator Manual



© **Doctor Web, 2023. All rights reserved**

This document is intended for information and reference purposes regarding the Dr.Web software discussed herein. This document is not a basis for exhaustive conclusions about the presence or absence of any functional and/or technical features in Dr.Web software and cannot be used to determine whether Dr.Web software meets any requirements, technical specifications and/or parameters, and other third-party documents.

This document is the property of Doctor Web and may be used solely for the personal purposes of the purchaser of the product. No part of this document may be reproduced, published or transmitted in any form or by any means, without proper attribution, for any purpose other than the purchaser's personal use.

Trademarks

Dr.Web, SpIDer Mail, SpIDer Guard, CureIt!, CureNet!, AV-Desk, KATANA and the Dr.WEB logo are trademarks and registered trademarks of Doctor Web in Russia and/or other countries. Other trademarks, registered trademarks and company names used in this document are the property of their respective owners.

Disclaimer

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

Dr.Web for UNIX Internet Gateways
Version 11.1
Administrator Manual
9/1/2023

Doctor Web Head Office

2-12A, 3rd str. Yamskogo polya, Moscow, Russia, 125124

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 that provide effective protection against malicious software and spam.

Doctor Web customers include home users around the world, government agencies, small businesses, and nationwide corporations.

Since 1992, Dr.Web anti-virus solutions have been known for their continuous excellence in malware detection and compliance with international information security standards.

The state certificates and awards received by Dr.Web solutions, as well as the worldwide use of our products, are the best evidence of exceptional trust in the company products.

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



Table of Contents

Introduction	8
Conventions and Abbreviations	9
About this Product	10
Dr.Web for UNIX Internet Gateways Main Functions	10
Structure of Dr.Web for UNIX Internet Gateways	12
Placing in Quarantine	20
File Permissions and Privileges	21
Operation Modes	22
System Requirements and Compatibility	25
Licensing	30
Installing and Uninstalling	31
Installing Dr.Web for UNIX Internet Gateways	32
Installing the Universal Package	33
Installing from the Command Line	35
Installing from the Repository	35
Upgrading Dr.Web for UNIX Internet Gateways	38
Updating Packages and Components	39
Upgrading to a New Product Version	40
Database Update without Internet Connection	43
Uninstalling Dr.Web for UNIX Internet Gateways	44
Uninstalling the Universal Package	44
Uninstalling from the Command Line	45
Uninstallation of Dr.Web for UNIX Internet Gateways Installed from the Repository	45
Additional Information	48
Dr.Web for UNIX Internet Gateways Packages and Files	48
Custom Component Installation and Uninstallation	51
Configuring Security Subsystems	57
Configuring SELinux Security Policies	57
Getting Started	61
Registration and Activation	61
Testing Product Operation	64
Integration with Squid Proxy Server	66
Protecting a Local Web Server	70



Using SplDer Gate in Proxy Mode	72
Brief Instructions	75
Dr.Web for UNIX Internet Gateways Components	80
Dr.Web ConfigD	80
Operating Principles	80
Command-Line Arguments	82
Configuration Parameters	83
Dr.Web Ctl	86
Command-Line Call Format	88
Usage Examples	117
Configuration Parameters	121
Dr.Web Web Management Interface	122
Managing the Components	124
Threats Management	125
Managing the Settings	127
Managing the Centralized Protection	131
Scanning Local Files	133
Dr.Web ICAPD	137
Operating Principles	137
Command Line Arguments	140
Configuration Parameters	141
HTTP Messages Processing in Lua	157
SplDer Gate	175
Operating Principles	176
Command-Line Arguments	177
Configuration Parameters	178
Dr.Web Firewall for Linux	180
Operating Principles	180
Command-Line Arguments	185
Configuration Parameters	186
Processing Connections in Lua	212
Dr.Web ClamD	221
Operating Principles	221
Command-Line Arguments	222
Configuration Parameters	222
Integration with External Applications	228



Dr.Web File Checker	231
Operating Principles	231
Command-Line Arguments	232
Configuration Parameters	233
Dr.Web Network Checker	235
Operating Principles	235
Command-Line Arguments	236
Configuration Parameters	237
Creating the Scanning Cluster	242
Dr.Web Scanning Engine	246
Operating Principles	246
Command-Line Arguments	247
Configuration Parameters	250
Dr.Web Updater	252
Operating Principles	252
Command-Line Arguments	253
Configuration Parameters	254
Dr.Web ES Agent	261
Operating Principles	261
Command-Line Arguments	262
Configuration Parameters	262
Dr.Web HTTPD	265
Operating Principles	265
Command-Line Arguments	266
Configuration Parameters	267
Description of the HTTP API	270
Dr.Web SNMPD	293
Operating Principles	293
Command-Line Arguments	294
Configuration Parameters	295
Integration with SNMP Monitoring Systems	299
Dr.Web SNMP MIB	307
Dr.Web MeshD	340
Operating Principles	340
Command-Line Arguments	344
Configuration Parameters	344



Dr.Web URL Checker	348
Operating Principles	348
Command-Line Arguments	349
Configuration Parameters	350
Dr.Web CloudD	351
Operating Principles	351
Command-Line Arguments	351
Configuration Parameters	352
Dr.Web LookupD	354
Operating Principles	354
Command-Line Arguments	355
Configuration Parameters	356
Dr.Web StatD	369
Operating Principles	369
Command-Line Arguments	369
Configuration Parameters	370
Appendices	372
Appendix A. Types of Computer Threats	372
Appendix B. Neutralizing Computer Threats	376
Appendix C. Technical Support	378
Appendix D. Dr.Web for UNIX Internet Gateways Configuration File	381
File Structure	381
Parameter Types	383
Rules for Traffic Monitoring	385
Appendix E. Generating SSL certificates	405
Appendix F. Known Errors	408
Appendix G. List of Abbreviations	460
Index	462



Introduction

Thank you for purchasing the Dr.Web for UNIX Internet Gateways solution. It offers reliable protection of your server from the spread of various types of [computer threats](#) by using the most advanced virus [detection and neutralization technologies](#). This improves the quality of services provided by the server.

This manual is intended to help administrators of the servers that run an UNIX OS-like, such as OS of the GNU/Linux family or FreeBSD, to install and use Dr.Web for UNIX Internet Gateways version 11.1.

Convention for File Paths

Real paths to files and components depend on the operating system. The document uses the following conventions for directories:

- *<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	<i><opt_dir></i>	/opt/drweb.com
	<i><etc_dir></i>	/etc/opt/drweb.com
	<i><var_dir></i>	/var/opt/drweb.com
FreeBSD	<i><opt_dir></i>	/usr/local/libexec/drweb.com
	<i><etc_dir></i>	/usr/local/etc/drweb.com
	<i><var_dir></i>	/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.



Conventions and Abbreviations

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

Convention	Comment
	An important note or instruction.
	A warning about possible errors or important notes that require special attention.
<i>Anti-virus network</i>	A new term or an emphasis on a term in descriptions.
<IP-address>	Placeholders.
Save	Names of buttons, windows, menu items and other program interface elements.
CTRL	Names of keyboard keys.
/home/user	Names of files and folders, code examples.
Appendix A	Cross-references to document chapters or internal hyperlinks to webpages.



Command-line commands, entered using a keyboard (in the terminal or a 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](#).



About this Product

In this section

- [Function](#)
- [Dr.Web for UNIX Internet Gateways Main Functions](#)
- [Structure of Dr.Web for UNIX Internet Gateways](#)
- [Placing in Quarantine](#)
- [File Permissions and Privileges](#)
- [Operation Modes](#)

Function

Dr.Web for UNIX Internet Gateways is designed to protect internet gateways and proxy running under UNIX (GNU/Linux, 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 (scan 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, 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 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 scan 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 Dr.Web products).

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, and so on). 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;
- *cloud-based threat detection technologies*, using the Dr.Web Cloud service that collects up-to-date information about recent threats and sends it to Dr.Web products.



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 custom scan of the specified objects only (separate directories or files that meet the specified criteria). 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.

All objects containing threats detected in the file system are registered in the permanently stored threats registry, except those threats that were detected in the autonomous copy mode.

The [Dr.Web Ctl](#) command-line tool included in Dr.Web for UNIX Internet Gateways, allows to scan for threats file systems of remote network hosts, that provide remote terminal access via SSH or Telnet.



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, and so on), 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 send data, Dr.Web for UNIX Internet Gateways 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.



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 scan 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. Sending of notifications on detected threats over SNMP to external monitoring systems and to the centralized protection server if Dr.Web for UNIX Internet Gateways operates in the [centralized protection mode](#), as well as to Dr.Web Cloud.
6. **Operation in the centralized protection mode** (when connected to the centralized 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, an internet service provider).

Structure of Dr.Web for UNIX Internet Gateways

Dr.Web for UNIX Internet Gateways is a product consisting of a set of components, where each component has its own set of functions. The components are separated into the following categories according to their objectives:

- [basic anti-virus components](#) which form Dr.Web for UNIX Internet Gateways core. In the absence of the components under this category, the product cannot scan files (and other data) for viruses and other threats;
- [threat search components](#). These components are used to solve Dr.Web for UNIX Internet Gateways basic tasks—detecting threats and potentially dangerous objects. In their operation the components falling under this category use basic anti-virus components;
- [service components](#), which solve the auxiliary anti-virus protection issues (anti-virus databases updates, centralized protection servers connection, common Dr.Web for UNIX Internet Gateways operation managing, and so on);
- [interface components](#), which provide (the user or third party applications) with the interface for Dr.Web for UNIX Internet Gateways.

Below is the list of Dr.Web for UNIX Internet Gateways components.



1. Basic Anti-virus Components

Component	Description
Dr.Web Virus-Finding Engine	<p>An anti-virus engine. Implements algorithms to detect viruses and malicious programs (by using a signature and heuristic analysis).</p> <p>Managed by Dr.Web Scanning Engine</p> <hr/> <p>Library file: <code>drweb32.dll</code>.</p> <p>Internal name displayed in the log file: <code>CoreEngine</code></p>
Dr.Web Scanning Engine	<p>A scan engine. This component loads Dr.Web Virus-Finding Engine and anti-virus databases.</p> <ul style="list-style-type: none">• Sends the contents of files and boot records to the anti-virus engine for scanning.• Manages a queue of the files to be scanned.• Cures threats to which this action is applicable. <p>Is operated by Dr.Web ConfigD or can operate autonomously.</p> <p>Used by Dr.Web File Checker and Dr.Web Network Checker components. Also can be used by Dr.Web MeshD components (in some operation modes) and by external (in relation to Dr.Web for UNIX Internet Gateways) applications specifically using Dr.Web Scanning Engine API</p> <hr/> <p>Executable file: <code>drweb-se</code>.</p> <p>Internal name displayed in the log: <code>ScanEngine</code></p>
Virus databases	<p>An automatically updated database of signatures of viruses and other threats, as well as of malicious software detection and neutralization algorithms.</p> <p>Used by Dr.Web Virus-Finding Engine and bundled with it</p>
Databases of web resource categories	<p>An automatically updated database containing a list of categorized web resources and being used to identify unwanted websites.</p> <p>Used by components that scan network activity of users and applications, such as SpIDer Gate, Dr.Web ICAPD</p>
Dr.Web File Checker	<p>A component for scanning file system objects and a quarantine manager.</p> <ul style="list-style-type: none">• Receives tasks from the threat scanning component on scanning files in the local (in relation to Dr.Web Scanning Engine) file system.• Surfs the file system directories according to the task, sends files for scanning to Dr.Web Scanning Engine and notifies client components about the scanning progress.• Deletes infected files, moves them to and restores them from quarantine, manages quarantine directories.



Component	Description
	<ul style="list-style-type: none">• Builds the cache and keeps it up-to-date. The cache contains information about previously scanned files to reduce the periodicity of rescanning files. <p>Used by components that scan file system objects</p> <hr/> <p>Executable file: <code>drweb-filecheck</code>.</p> <p>Internal name displayed in the log: <code>FileCheck</code></p>
Dr.Web Network Checker	<p>A network data scanning agent.</p> <ul style="list-style-type: none">• Used to send data to the scan engine. The data is sent by components of the product over the network (such as Dr.Web ClamD, SpIDer Gate, Dr.Web ICAPD).• Allows Dr.Web for UNIX Internet Gateways to manage distributed file scanning: to receive/transmit files for scanning from/to remote hosts. For that purpose, the remote hosts must feature an installed and running Dr.Web for UNIX 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). <p>If partner hosts that can receive data for scanning are present on the network, the components that use Dr.Web Network Checker for scanning may not use local Dr.Web Scanning Engine. Thus, Dr.Web Scanning Engine, Dr.Web Virus-Finding Engine and anti-virus databases may be absent.</p> <p>For security reasons, files are transmitted over the network using SSL</p> <hr/> <p>Executable file: <code>drweb-netcheck</code>.</p> <p>Internal name displayed in the log: <code>NetCheck</code></p>
Dr.Web URL Checker	<p>A component for analyzing if a URL falls under potentially dangerous/unwanted categories.</p> <hr/> <p>Executable file: <code>drweb-urlcheck</code>.</p> <p>Internal name displayed in the log: <code>UrlCheck</code></p>
Dr.Web MeshD	<p>A component that connects Dr.Web for UNIX Internet Gateways to the local cloud, which allows Dr.Web for UNIX products to exchange updates, results of file scanning, transmit files to each other for scanning, as well as provide scan engine services directly.</p> <p>If the product includes this component, a local cloud to which this component is connected, as well as hosts providing scan engine services, Dr.Web Scanning Engine, Dr.Web Virus-Finding Engine and anti-virus databases may be absent</p> <hr/> <p>Executable file: <code>drweb-meshd</code>.</p> <p>Internal name displayed in the log: <code>MeshD</code></p>



2. Threat Search Components

Component	Description
Dr.Web ICAPD	<p>An ICAP server analyzing requests and network traffic passing through HTTP proxies supporting ICAP (such as Squid).</p> <p>It prevents transmitting infected files and accessing network hosts falling under unwanted categories of internet resources and added to black lists created by the system administrator. If the access to external servers is forbidden or transmitted data contains a threat, the proxy server is instructed to return to the user a special page informing that it is impossible to access the requested resource or download the infected file.</p> <p>Uses Dr.Web Network Checker to scan data received from the proxy server</p> <hr/> <p>Executable file: <code>drweb-icapd</code>.</p> <p>Internal name displayed in the log: <code>ICAPD</code></p>
SpIDer Gate	<p>A component for monitoring network traffic and URLs.</p> <p>It is designed to scan data downloaded from the network to the local host and transmitted from it to the external network for threats. The component also prevents connections with the network hosts added to the unwanted categories of web resources or black lists created by the system administrator.</p> <p>Uses Dr.Web Network Checker to scan received data.</p> <div style="background-color: #e6f2e6; padding: 10px; border: 1px solid #ccc;"> It is included only in the distributions designed for GNU/Linux OSes.</div> <hr/> <p>Executable file: <code>drweb-gated</code>.</p> <p>Internal name displayed in the log: <code>GateD</code></p>
Dr.Web Firewall for Linux	<p>A network connections monitor.</p> <p>It is used by SpIDer Gate and provides connection routing for applications that operate on the server for scanning the transmitted network traffic.</p> <div style="background-color: #e6f2e6; padding: 10px; border: 1px solid #ccc;"> It is included only in the distributions designed for GNU/Linux OSes.</div> <hr/> <p>Executable file: <code>drweb-firewall</code>.</p> <p>Internal name displayed in the log file: <code>LinuxFirewall</code></p>



3. Service Components

Component	Description
Dr.Web CloudD	<p>A component for interaction with Dr.Web Cloud.</p> <p>Sends URLs visited by the user and information about the scanned files to the Dr.Web Cloud service to scan them for threats not yet covered by virus databases</p> <hr/> <p>Executable file: <code>drweb-cloudd</code>.</p> <p>Internal name displayed in the log: <code>CloudD</code></p>
Dr.Web ConfigD	<p>Dr.Web for UNIX Internet Gateways configuration daemon.</p> <ul style="list-style-type: none">• Starts and stops other product components depending on the settings.• 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 information 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 <hr/> <p>Executable file: <code>drweb-configd</code>.</p> <p>Internal name displayed in the log file: <code>ConfigD</code></p>
Dr.Web ES Agent	<p>The centralized protection agent. Ensures product operation in the centralized protection and mobile modes.</p> <ul style="list-style-type: none">• Provides connection between the product and the centralized protection server, receives a license key file, updates of the virus databases and anti-virus engine.• Sends to the server information on the components included in Dr.Web for UNIX Internet Gateways and their status as well as statistics of virus events <hr/> <p>Executable file: <code>drweb-esagent</code>.</p> <p>Internal name displayed in the log: <code>ESAgent</code></p>
Dr.Web LookupD	<p>A component for retrieving data from external data sources.</p> <p>Retrieves data from external data sources (directory services, files, relative databases, and so on) to be used in rules of traffic monitoring</p> <hr/> <p>Executable file: <code>drweb-lookupd</code>.</p> <p>Internal name displayed in the log: <code>LookupD</code></p>
Dr.Web StatD	<p>A component for storing Dr.Web for UNIX Internet Gateways component operation events.</p>



Component	Description
	<p>Receives and stores events of the product components (such as abnormal termination, threat detection, and so on)</p> <hr/> <p>Executable file: <code>drweb-statd</code>.</p> <p>Internal name displayed in the log file: <code>StatD</code></p>
Dr.Web Updater	<p>An updating component.</p> <p>Downloads from Doctor Web servers updates of the virus databases and databases of web resource categories, the anti-virus engine.</p> <p>The updates can be downloaded automatically, according to the schedule, and on user demand (via Dr.Web Ctl or management web interface)</p> <hr/> <p>Executable file: <code>drweb-update</code>.</p> <p>Internal name displayed in the log: <code>Update</code></p>

4. Interface Components

Component	Description
Dr.Web HTTPD	<p>Dr.Web for UNIX Internet Gateways management web server.</p> <p>Provides a custom HTTP API for managing Dr.Web for UNIX Internet Gateways components.</p> <p>The specified API is used by the management web interface (must be installed separately).</p> <p>For security reasons, the component uses HTTPS to connect to the web interface.</p> <p>Uses Dr.Web Network Checker to transmit data for scanning to Dr.Web Scanning Engine</p> <hr/> <p>Executable file: <code>drweb-httpd</code>.</p> <p>Internal name displayed in the log: <code>HTTPD</code></p>
Dr.Web Web Management Interface	<p>Management Web Interface.</p> <p>The interface can be accessed using any browser on a local or remote host. The web interface enables the product not to use third-party web servers (such as Apache HTTP Server) or remote administration tools, such as Webmin.</p> <p>The component functionality is provided by Dr.Web HTTPD component</p>
Dr.Web Ctl	<p>A tool for managing Dr.Web for UNIX Internet Gateways from the command line.</p> <p>Allows the user to start file scanning, view and manage quarantined objects, start a virus database update procedure, connect Dr.Web for</p>



Component	Description
	<p>UNIX Internet Gateways to or disconnect it from the centralized protection server, view and configure product parameters</p> <hr/> <p>Executable file: <code>drweb-ctl</code>.</p> <p>Internal name displayed in the log: <code>ctl</code></p>
Dr.Web SNMPD	<p>An SNMP agent.</p> <p>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 components and to collect statistics on threat detection and neutralization.</p> <p>Supports SNMP v2c and v3</p> <hr/> <p>Executable file: <code>drweb-snmpd</code>.</p> <p>Internal name displayed in the log: <code>SNMPD</code></p>
Dr.Web ClamD	<p>A component emulating interface of the <code>clamd</code>, anti-virus daemon (the component of ClamAV® anti-virus).</p> <p>Allows all applications supporting ClamAV® to use Dr.Web for UNIX Internet Gateways transparently for anti-virus scanning.</p> <p>Depending on the mode, uses Dr.Web File Checker or Dr.Web Network Checker to transmit data for scanning to Dr.Web Scanning Engine</p> <hr/> <p>Executable file: <code>drweb-clamd</code>.</p> <p>Internal name displayed in the log: <code>ClamD</code></p>



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

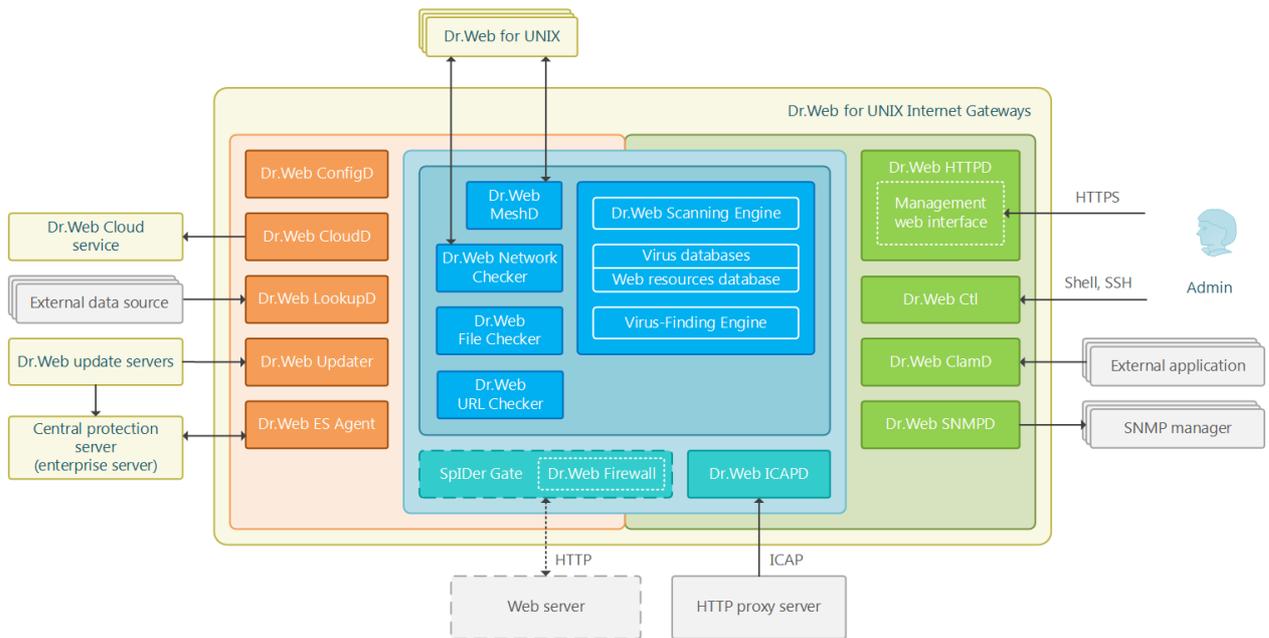


Figure 1. The structure of Dr.Web for UNIX Internet Gateways

In this scheme, the following notations are used:

	—Dr.Web for UNIX Internet Gateways as a whole and external Dr.Web applications not included in the solution
	—programs external to Dr.Web for UNIX Internet Gateways and products that integrate with it
	—the service components that perform particular anti-virus protection tasks (anti-virus databases updates, centralized protection servers connection, overall coordination of operations, and so on)
	—components that provide (the user or third party applications) with the interface to manage Dr.Web for UNIX Internet Gateways
	—components used for anti-virus scanning
	—basic anti-virus components that form the Dr.Web for UNIX Internet Gateways core. Used by the components that perform data and files scans

Components marked with a dashed line can be missing depending on the Dr.Web for UNIX Internet Gateways distribution and usage.

For details on Dr.Web for UNIX Internet Gateways components, refer to [Dr.Web for UNIX Internet Gateways Components](#).



Placing in Quarantine

Quarantine directories of Dr.Web for UNIX Internet Gateways 11.1 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 (is not included in 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, and so on), 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.  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.



The Dr.Web Scanning Engine scan engine cannot scan file which size exceeds 4 GB (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 *centralized protection server*. Operation in the *centralized 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 the centralized protection mode (enterprise mode)*, protection of the computer is managed by the centralized 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 the centralized protection mode is received from the centralized protection server. The demo key file stored on the local computer, if any, is not used. Statistics on virus events together with information on Dr.Web for UNIX Internet Gateways operation are sent to the centralized protection server. Updates to virus databases are also received from the centralized protection server.
- *In the 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 and a license key file received from the centralized protection server. You can switch to the mobile mode only if it is allowed in the centralized protection server settings.

Centralized Protection Concept

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

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

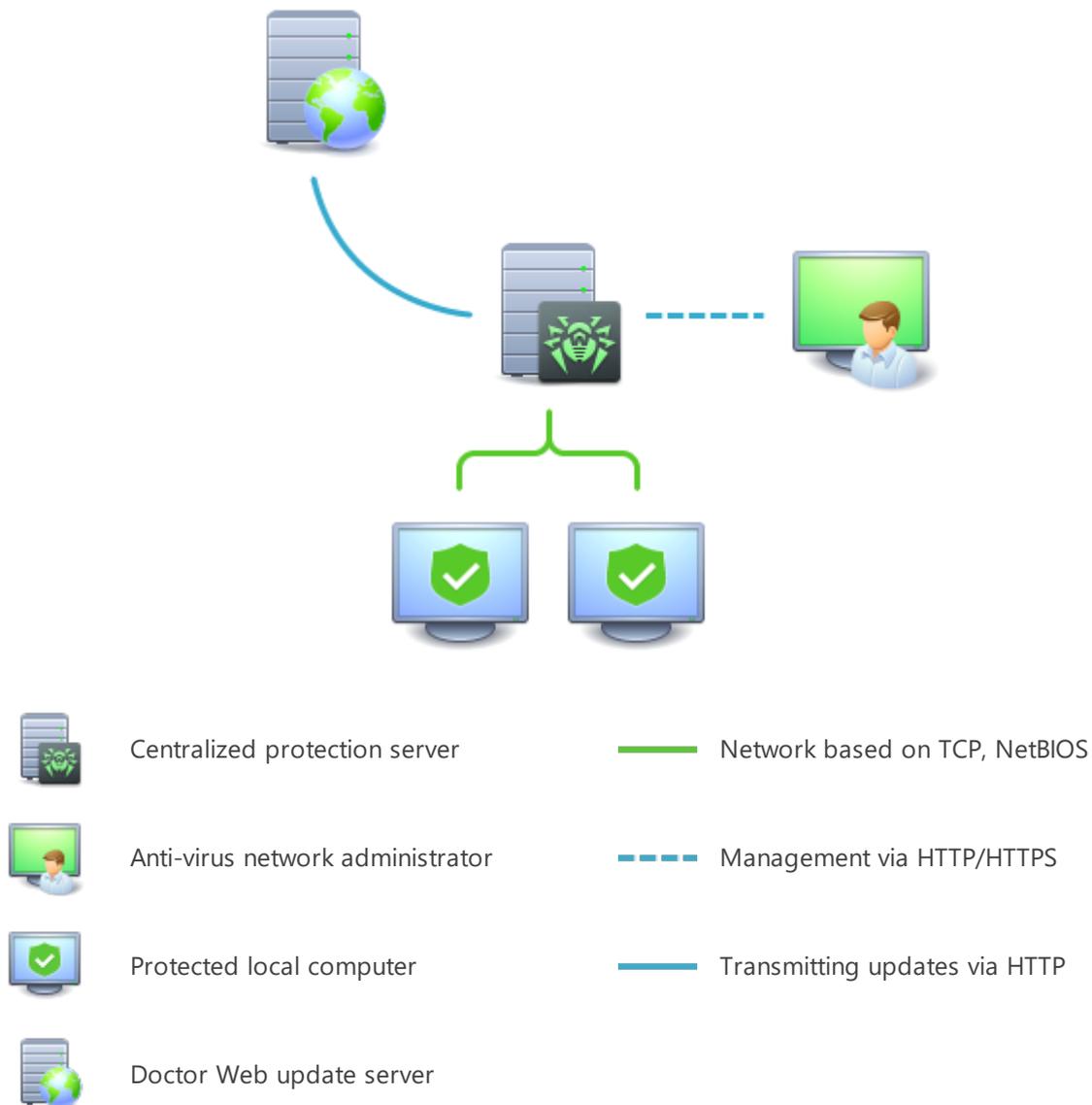


Figure 2. Logical structure of the Anti-virus Network

Local computers are updated and configured from the *centralized protection server*. The stream of instructions, data and statistics in the anti-virus network goes also through the centralized 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 centralized protection server from Doctor Web update servers.

Local anti-virus components are configured and managed from the centralized protection server according to commands received from anti-virus network administrators. Administrators manage centralized protection servers and topology of anti-virus networks (for example,



validate connections to the centralized 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 the centralized 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.

Centralized protection mode allows exporting and saving operation reports using the centralized protection center. Reports can be exported and saved in the following formats: HTML, CSV, PDF, and XML.

Connection to the centralized protection server

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



For the verification of the centralized protection server the certificate corresponding to the unique public key of the server is used. By default, the Dr.Web ES Agent centralized protection agent will not allow you to connect to the server unless you specify the certificate file. The certificate file must first be obtained from the administrator of the anti-virus network served 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 centralized protection server, you can switch the product into the mobile mode or switch it back into the centralized protection mode. Switching the mobile mode on or off is accomplished with the help of the `MobileMode` [configuration parameter](#) of the [Dr.Web ES Agent](#) component.



Dr.Web for UNIX Internet Gateways can switch to the mobile mode only if it is allowed in the settings on the centralized protection server.

Disconnecting from an Anti-Virus Network

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



System Requirements and Compatibility

In this section

- [System Requirements](#)
- [List of Supported Operating System Versions](#)
- [Additional Packages and Components](#)
- [Disclaimer](#)
- [Supported HTTP Proxy Servers](#)
- [Compatibility with Security Subsystems](#)

System Requirements

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

Component	Requirement
Platform	Processors of the following architectures and command systems are supported: <ul style="list-style-type: none">• Intel/AMD: 32-bit (<i>IA-32, x86</i>); 64-bit (<i>x86-64, x64, amd64</i>)• ARM64• E2K (<i>Elbrus</i>)• IBM POWER (<i>ppc64el</i>)
RAM	At least 500 MB of free RAM (1 GB or more is recommended)
Free disk space	At least 2 GB of free disk space on a volume where the product directories are located
Operating system	GNU/Linux (based on kernel version 2.6.37 or later, using <code>glibc</code> library 2.13 or later, <code>systemd</code> initialization system ver. 209 or later), FreeBSD. The supported operating system versions are listed below. The operating system must support the PAM authentication mechanism
Other	The following valid network connections: <ul style="list-style-type: none">• valid Internet connection to enable updates for virus databases and Dr.Web components;• when operating in the centralized protection mode, connection to the server on the local network is enough; connection to the Internet is not required



For the correct operation of the component Dr.Web Firewall for Linux, 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 version kit.

To ensure the correct operation Dr.Web for UNIX Internet Gateways, open the following ports:

Purpose	Direction	Port numbers
To receive updates	outgoing	80
To connect to the Dr.Web Cloud service	outgoing	2075 (including those for UDP), 3010 (TCP), 3020 (TCP), 3030 (TCP), 3040 (TCP)

List of Supported Operating System Versions

• GNU/Linux

Platform	Supported GNU/Linux versions
x86_64	<ul style="list-style-type: none">• Astra Linux Special Edition 1.5 (with cumulative patch 20201201SE15), 1.6 (with cumulative patch 20200722SE16), 1.7• Astra Linux Common Edition (Orel) 2.12• Debian 9, 10• Fedora 31, 32• CentOS 7, 8• Ubuntu 18.04, 20.04, 22.04• ALT Workstation 9, 10• ALT Server 9, 10• ALT 8 SP• RED OS 7.2 MUROM, RED OS 7.3 MUROM• GosLinux IC6• SUSE Linux Enterprise Server 12 SP3• Red Hat Enterprise Linux 7, 8
x86	<ul style="list-style-type: none">• CentOS 7• Debian 10



Platform	Supported GNU/Linux versions
	<ul style="list-style-type: none">• ALT Workstation 9, 10• ALT 8 SP
ARM64	<ul style="list-style-type: none">• Ubuntu 18.04• CentOS 7, 8• ALT Workstation 9, 10• ALT Server 9, 10• ALT 8 SP• Astra Linux Special Edition (Novorossiysk) 4.7
E2K	<ul style="list-style-type: none">• Astra Linux Special Edition (Leningrad) 8.1 (with cumulative patch 8.120200429SE81)• ALT 8 SP• Elbrus-D MCST 1.4• GS CS Elbrus 8.32 TVGI.00311-28
ppc64el	<ul style="list-style-type: none">• CentOS 8;• Ubuntu 20.04



In ALT 8 SP, Astra Linux Special Edition (Novorossiysk) 4.11, Elbrus-D MCST 1.4 and GosLinux IC6 mandatory access control is not supported.

For other GNU/Linux versions that meet the abovementioned requirements full compatibility with Dr.Web for UNIX Internet Gateways is not guaranteed. If a compatibility issue occurs, contact [technical support](#).

• FreeBSD

Platform	Supported FreeBSD versions
x86	11, 12, 13
x86_64	11, 12, 13



For FreeBSD OS, Dr.Web for UNIX Internet Gateways can be installed only from the [universal package](#).

Additional Packages and Components

Dr.Web for UNIX Internet Gateways does not require installation of additional packages and OS components (except for the protected server software, see below).



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

If you encounter any problem with installation of additional packages and components, refer to the documentation of your operating system version.

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, Red Hat 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.
- If the used OS includes the version of NetFilter less than 1.4.15, SpIDer Gate can 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 “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 the [internet barrier](#) and [transparent proxy](#), 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 the 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 Dr.Web for UNIX Internet Gateways installation. The license contains information on the user and the vendor as well as usage parameters of the purchased copy, including:

- list of components licensed to the user;
- Dr.Web for UNIX Internet Gateways license period;
- other restrictions (for example, number of computers on which the purchased copy is allowed for use).

For evaluation purposes users can also activate the *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 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 Dr.Web for UNIX Internet Gateways by connecting it to the centralized 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 centralized protection server.



Installing and Uninstalling

In this section

- [Installing Dr.Web for UNIX Internet Gateways](#)
- [Upgrading Dr.Web for UNIX Internet Gateways](#)
- [Uninstalling Dr.Web for UNIX Internet Gateways](#)
- [Configuring Security Subsystems](#)
- Additional information:
 - [Dr.Web for UNIX Internet Gateways Packages and Files](#)
 - [Custom Component Installation and Uninstallation](#)

This section describes how to install and uninstall the Dr.Web for UNIX Internet Gateways version 11.1. 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 Dr.Web for UNIX Internet Gateways components (for example, to resolve errors that occurred during the course of the Dr.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 Dr.Web for UNIX Internet Gateways.

To perform these procedures, 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.



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 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 Dr.Web for UNIX Internet Gateways you want to install. If the product version that you plan on installing is newer than the installed product version, before installation, it is necessary to [upgrade](#) the installed Dr.Web for UNIX Internet Gateways to the version of the product you want to install additionally.

For FreeBSD OS, Dr.Web for UNIX Internet Gateways can be installed only from the [universal package](#).

Installing Dr.Web for UNIX Internet Gateways

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

1. From the Doctor Web 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 graphical desktop mode, you will need to have a terminal emulator available).
2. Download the [native packages](#) from the corresponding package repository of Doctor Web.



For FreeBSD OS, Dr.Web for UNIX Internet Gateways can be installed only from the [universal package](#).



In OSs using the outdated versions of the package manager (for instance, ALT 8 SP) it is recommended to install the Dr.Web for UNIX Internet Gateways [universal package](#).

During the installation (as well as from universal .run package, as from native packages using package manager), on `root@localhost` email address messages containing installation results are sent.

After you installed Dr.Web for UNIX Internet Gateways by any of the mentioned means, you can [uninstall](#) or [update](#) it if there are fixes for its components available or if a new Dr.Web for UNIX Internet Gateways version is released. If required, you can also [configure security subsystems](#) of GNU/Linux for correct operation of Dr.Web for UNIX Internet Gateways. If there is a problem



with functioning of any individual components, you can perform their [custom installation and uninstallation](#), without uninstalling Dr.Web for UNIX Internet Gateways.

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](#) Dr.Web for UNIX Internet Gateways to the centralized protection server. For details, refer to [Licensing](#). Unless you do it, anti-virus protection functions will be switched off. In addition, in some cases it is necessary to customize the basic functionality of the installed Dr.Web for UNIX Internet Gateways, as described in the [Getting Started](#) section.

Installing the Universal Package

Dr.Web for UNIX Internet Gateways universal package is distributed as an installation file named `drweb-<version>-av-igw-<OS>-<platform>.run`, where `<OS>` is a type of UNIX-like OS, `<Platform>` is the platform for which Dr.Web for UNIX Internet Gateways is intended (for 32-bit platforms—`x86`, for 64-bit platforms—`amd64`, `arm64` and `e2s`). For example:

```
drweb-11.1.0-av-igw-linux-x86.run
```



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 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 command:

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

4. Execute the archive using the 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 instructions.
6. You can also start the installation program in a silent mode by executing the 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).



Using this option means that you accept the terms of the Dr.Web License Agreement. The License Agreement 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, [uninstall](#) Dr.Web for UNIX Internet Gateways 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 your GNU/Linux distribution features SELinux, the installation process can be interrupted by the security subsystem. If such a situation occurs, set SELinux to the *Permissive* mode with the 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 the `<file_name>.run` file, from which the installation was performed, to reinstall Dr.Web for UNIX Internet Gateways or its components without the need to update its version.



Installing from the 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 installation, enter `Yes` or `Y` in response to the “Do you want to continue?” question. To exit the installer, enter `No` or `N`. In this case, installation is 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.



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 Dr.Web components, will be displayed on the screen.
5. If installation completes successfully, a message is displayed that informs you on how to manage Dr.Web for UNIX Internet Gateways operation.

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.

Installing from the Repository

Dr.Web for UNIX Internet Gateways 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 package manager, you can install the product from native packages as you install any other programs from the operating system 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).

For the FreeBSD OS, Dr.Web for UNIX Internet Gateways can be installed only from the [universal package](#).



See below the procedures for the following OS (package managers):

- [Debian, Mint, Ubuntu \(apt\)](#),
- [ALT Linux, PCLinuxOS \(apt-rpm\)](#),
- [Mageia, OpenMandriva Lx \(urpmi\)](#),
- [Red Hat Enterprise Linux, Fedora, CentOS \(yum, dnf\)](#),
- [SUSE Linux \(zypper\)](#).

Debian, Mint, Ubuntu (apt)

To install Dr.Web for UNIX Internet Gateways from the repository

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 command:

```
# apt-key adv --keyserver hkp://keyserver.ubuntu.com:80 --recv-keys
8C42FC58D8752769
```

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

```
deb http://repo.drweb.com/drweb/debian 11.1 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/drweb-repo11.1.deb>.

3. To install Dr.Web for UNIX Internet Gateways from the repository, use the 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)

To install Dr.Web for UNIX Internet Gateways from the repository

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

```
rpm http://repo.drweb.com/drweb/altlinux 11.1/<arch> drweb
```

where `<arch>`—representation of the package architecture:

- for the 32-bit version: `i386`;



- for the AMD64 architecture: `x86_64`;
- for the ARM64 architecture: `aarch64`;
- for the E2K architecture: `e2s`.

2. To install Dr.Web for UNIX Internet Gateways from the repository, use the 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)

To install Dr.Web for UNIX Internet Gateways from the repository

1. Connect the repository using the command:

```
# urpmi.addmedia drweb https://repo.drweb.com/drweb/linux/11.1/<arch>/
```

where `<arch>`—representation of the package architecture:

- for the 32-bit version: `i386`;
- for the 64-bit version: `x86_64`.

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

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

You can also use alternative package managers (for example, `rpm-drake`) to install the product.

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

To install Dr.Web for UNIX Internet Gateways from the repository

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

```
[drweb]
name=DrWeb-11.1
baseurl=https://repo.drweb.com/drweb/linux/11.1/$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/drweb-repo11.1.rpm>.

2. To install Dr.Web for UNIX Internet Gateways from the repository, use the 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)

To install Dr.Web for UNIX Internet Gateways from the repository

1. To add the repository, use the command:

```
# zypper ar https://repo.drweb.com/drweb/linux/11.1/\$basearch/ drweb
```

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

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

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

Upgrading Dr.Web for UNIX Internet Gateways

Dr.Web for UNIX Internet Gateways has two update modes.

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



Dr.Web for UNIX Internet Gateways provides the ability [to update virus databases and the anti-virus engine](#) even without access to the internet on the protected server.

Updating Packages and Components

After installation of Dr.Web for UNIX Internet Gateways 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), for operation with Dr.Web packages, an individual version of package managers `zypper` is used. It is automatically installed during the Dr.Web for UNIX Internet Gateways 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 commands:

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



In the FreeBSD OS 11.x for amd64 a repository update error may occur upon using the `zypper` manager for updates. In this case, install the `compat10x-amd64` support package and try again.

To install the package, use the command:

```
# pkg install compat10x-amd64
```

- 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 version of Dr.Web for UNIX Internet Gateways is released, packages with its components are put into the section of the Dr.Web repository corresponding to the new version. In this case, an update requires switching of the package manager to a new Dr.Web repository section (refer to [Upgrading to a New Product Version](#)).



Upgrading to a New Product Version

In this section

- [Introductory Remarks](#)
- [Installing Universal Package for an Upgrade](#)
- [Upgrading from the Repository](#)
- [Key File Transfer](#)
- [Restoring Connection to the Centralized Protection Server](#)

Introductory Remarks



Before you upgrade to a new version, make sure that your server meets the [system requirements](#) of the new version, including that the necessary programs must be installed.

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



To identify how the product version was installed, check whether the Dr.Web for UNIX Internet Gateways executable directory contains the `uninst.sh` uninstallation script. If so, the current version was installed from the universal package; otherwise it was installed from the repository.

For FreeBSD OS, Dr.Web for UNIX Internet Gateways can be installed only from the [universal package](#).

If you cannot update Dr.Web for UNIX Internet Gateways 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 11.1. For additional information, see User manual for your current version of Dr.Web for UNIX Internet Gateways.



If the current version of Dr.Web for UNIX Internet Gateways is operating in the [centralized protection](#) mode, it is recommended that you record the address of the centralized 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 command:

```
$ drweb-ctl appinfo
```

In the output of this command find the following line:

```
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 certificate file.

In case there are any problems with finding out the parameters of the connection that you are currently using, refer to the Administrator Manual for the Dr.Web for UNIX Internet Gateways 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 11.1 from the [universal package](#). If an automatic update is impossible, during the installation of the new version, you will get an offer to remove the components of the older version of Dr.Web for UNIX Internet Gateways installed on your computer automatically.



If during the update process you need to remove the installed Dr.Web for UNIX Internet Gateways version, and there are multiple Dr.Web 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 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`.

Upgrading from the Repository



You cannot upgrade Dr.Web for UNIX Internet Gateways 6.0.2 to version 11.1 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 installed from the Doctor Web repository

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

• RPM packages (yum, dnf)

1. Change the repository (from the package repository of your current version to the package repository 11.1).



You can find the name of the repository that stores 11.1 packets in the [Installing from the Repository](#) section. For details on how to change repositories, refer to help guides of your operating system distribution.

2. Install the new version using the command:

```
# yum update
```

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

```
# dnf update
```



If during the update of packages there is an error, uninstall Dr.Web for UNIX Internet Gateways and install it again. If necessary, see sections [Uninstallation of Dr.Web for UNIX Internet Gateways 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 repository (from the package repository of your current version to the package repository 11.1).
2. Upgrade the Dr.Web for UNIX Internet Gateways packages by entering the commands:

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



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 Dr.Web for UNIX Internet Gateways, your current license [key file](#) (if you have one) will be automatically transferred and installed to the correct location required for the new version.



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 Centralized Protection Server

If it is possible, your connection to the centralized protection server will be restored automatically after the upgrade (if the product had been connected to the centralized 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 [command](#):

```
$ drweb-ctl esconnect <address> --Certificate <path to the certificate file>
```

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

Database Update without Internet Connection

In highly secure environments where internet connection is blocked or limited, it is possible to *update virus bases offline*. You need to download updates to a computer connected to the internet, copy them to a USB drive or local network share and then install them to another computer (which is not connected to the internet). The update procedure has to be run from the command line.

To get the updates

1. Run the command on a computer connected to the internet:

```
$ drweb-ctl update --Path <a path to a directory to store updates>
```

2. Copy the downloaded updates to a USB drive or local network share.
3. Mount the local network share or removable drive on the computer to be updated. If the updates are from the USB drive, run the commands:

```
# mkdir /mnt/usb  
# mount <a path to the device> /mnt/usb
```



4. Apply the updates with the command:

```
$ drweb-ctl update --From /mnt/usb
```

Uninstalling Dr.Web for UNIX Internet Gateways

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

1. [Start the uninstaller](#) to uninstall the universal package.
2. [Uninstall the packages](#) installed from the Doctor Web repository with the help of the system 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).



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 Dr.Web for UNIX Internet Gateways 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).



Root privileges are required to start the uninstallation tool in silent mode. To elevate the privileges, you can use the `su` and `sudo` commands.



When uninstalling the universal package in OSs using the outdated versions of the package manager (for instance, ALT 8 SP), you may see messages of the following type:

```
/etc/init.d/drweb-configd: No such or directory
```

These messages do not affect the functioning of the system. The uninstallation procedure is performed correctly.

Uninstalling from the Command Line

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

1. To initiate the removal, enter `Yes` or `Y` in response to the “Do you want to continue?” request. To exit the uninstaller, type `No` or `N`. In this case, removal of Dr.Web products will be canceled.
2. An automatic uninstallation procedure of all installed Dr.Web packages will be launched after you confirm it. During this procedure, information about the removal process will be displayed on the screen and entered into the uninstallation log.
3. Once the process is completed, the uninstallation program will automatically terminate.

Uninstallation of Dr.Web for UNIX Internet Gateways Installed from the Repository

See below the procedures for the following OS (package managers):

- [Debian, Mint, Ubuntu \(apt\)](#);
- [ALT Linux, PCLinuxOS \(apt-rpm\)](#);
- [Mageia, OpenMandriva Lx \(urpmi\)](#);
- [Red Hat Enterprise Linux, Fedora, CentOS \(yum, dnf\)](#);
- [SUSE Linux \(zypper\)](#).



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 together, enter the command:

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

If you need to uninstall the root meta-package together with all dependencies, use the `--autoremove` option:

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

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

```
# apt-get autoremove
```



Special Aspects of Uninstallation

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 products). 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). 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.



On ALT 8 SP you may see the following messages appear when the universal package is being uninstalled:

```
/etc/init.d/drweb-configd: No such or directory
```

These messages do not affect the functioning of the system. The uninstallation procedure is performed correctly.

Mageia, OpenMandriva Lx (urpme)

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

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

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

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



Special Aspects of Uninstallation

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).

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, `rpmrake`) to uninstall packages.

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

To uninstall all the installed Dr.Web packages, enter the 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*
```



Special Aspects of Uninstallation

These variants of the command uninstall all the packages whose name starts with "drweb" (the standard name prefix for Dr.Web products).

These commands uninstall 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 command:

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

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

```
# zypper remove drweb*
```



Special Aspects of Uninstallation

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 products). 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

Dr.Web for UNIX Internet Gateways Packages and Files

Packages

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

Package	Contents
drweb-bases	Virus database files



Package	Contents
drweb-boost	Boost libraries
drweb-clamd	Files of the Dr.Web ClamD component
drweb-cloudd	Files of the Dr.Web CloudD component
drweb-common	<p>The main configuration file—<code>drweb.ini</code>, main libraries, documentation, hierarchy of the Dr.Web for UNIX Internet Gateways directories, and utility for collection information on product configuration and system environment.</p> <p>During the installation of this package, a user named <code>drweb</code> and a group named <code>drweb</code> are created</p>
drweb-configd	Dr.Web ConfigD Files
drweb-configure	Files of auxiliary tool for Dr.Web for UNIX Internet Gateways configuration
drweb-ctl	Dr.Web Ctl files
drweb-documentation	Dr.Web for UNIX products documentation files in HTML format
drweb-dws	Files of a database of web resource categories
drweb-engine	Dr.Web Virus-Finding Engine scan engine files
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 Documents
drweb-internet-gateways	The root meta-package of Dr.Web for UNIX Internet Gateways
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



Package	Contents
drweb-libs	Main libraries files
drweb-lookupd	Files of the Dr.Web LookupD component
drweb-lua	Files of the Lua interpreter used by Dr.Web for UNIX Internet Gateways components designed for monitoring network connections
drweb-netcheck	Files of the Dr.Web Network Checker component
drweb-openssl	OpenSSL libraries
drweb-protobuf	Google 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

In the section [Custom Component Installation and Uninstallation](#) there are typical component sets for a custom installation that provide solutions for typical tasks of Dr.Web for UNIX Internet Gateways.

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 Dr.Web for UNIX Internet Gateways directories:

Directory	Contents
<ul style="list-style-type: none">• For GNU/Linux: <code>/etc/init.d/</code>• For FreeBSD: <code>/usr/local/etc/rc.d/</code>	<code>drweb-configd</code> script for the Dr.Web ConfigD daemon
<code><etc_dir></code>	The <code>drweb.ini</code> configuration file and the <code>drweb32.key</code> key file. In addition, it contains:
<code>certs/</code>	– files of certificates in use.
<code><opt_dir>/</code>	Main directory of Dr.Web for UNIX Internet Gateways. It contains:
<code>bin/</code>	– executable files of all product components (except for Dr.Web Virus-Finding Engine);
<code>include/</code>	– heading files of libraries in use;



Directory	Contents
lib/	– libraries in use;
man/	– system help files: man;
share/	– auxiliary product files, including:
doc/	▫ product documentation (readme files, license agreement and administrator guide if packages are already installed),
drweb-bases/	▫ files of Dr.Web virus databases (source files supplied during installation),
scripts/	▫ auxiliary scripts files.
<var_dir>/	Auxiliary and temporary files of, including:
bases/	– files of Dr.Web virus databases (the updated version);
cache/	– cache of updates;
drl/	– the lists of update servers in use;
dws/	– files of a database of web resource categories;
lib/	– the Dr.Web Virus-Finding Engine scan engine as a dynamic-link library (drweb32.dll) and the settings for working in the centralized 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

In this section

- [Typical Component Kits for a Custom Installation](#)
- Dr.Web for UNIX Internet Gateways Component Installation and Uninstallation:
 - [installed from the repository](#)
 - [installed from the universal package](#)

If necessary, you can choose to install or uninstall only certain Dr.Web for UNIX Internet Gateways components by installing or uninstalling the respective [packages](#). Perform custom component installation or uninstallation 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 Dr.Web for UNIX Internet Gateways with the limited functionality, instead of installation of the 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 Dr.Web for UNIX Internet Gateways 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	Packages for Installation	Will be Installed
Minimum kit for console scanning	<ul style="list-style-type: none">• drweb-filecheck,• drweb-se	<ul style="list-style-type: none">• Dr.Web ConfigD,• Dr.Web Ctl,• Dr.Web File Checker,• Dr.Web Scanning Engine,• Dr.Web Updater,• Virus databases
Suite for the emulation ClamAV (clamd)	<ul style="list-style-type: none">• drweb-clamd,• drweb-se	<ul style="list-style-type: none">• Dr.Web ClamD,• Dr.Web ConfigD,• Dr.Web Ctl,• Dr.Web File Checker,• Dr.Web Network Checker,• Dr.Web Scanning Engine,• Dr.Web Updater,• Virus database
Suite for checking the access to websites using a proxy server via the ICAP protocol (without the anti-virus traffic scanning)	<ul style="list-style-type: none">• drweb-icapd	<ul style="list-style-type: none">• Dr.Web ConfigD,• Dr.Web Ctl,• Dr.Web ICAPD,• Dr.Web Updater,• Dr.Web URL Checker,• 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).	<ul style="list-style-type: none">• drweb-icapd,• drweb-netcheck,• drweb-se	<ul style="list-style-type: none">• Dr.Web ConfigD,• Dr.Web Ctl,• Dr.Web ICAPD,• Dr.Web Network Checker,• Dr.Web Scanning Engine, *• Dr.Web Updater,• Dr.Web URL Checker,



Custom Component Kit	Packages for Installation	Will be Installed
 <p>The package <code>drweb-se</code> 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.</p>		<ul style="list-style-type: none">• Database of web resource, categories,• Virus database *
<p>Suite for a local scanning of HTTP connections.</p>  <p>If the anti-virus scanning of connections is not required, packages <code>drweb-netcheck</code> and <code>drweb-se</code> are not required for installation. The package <code>drweb-se</code> 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 <code>drweb-dws</code> could be skipped for installation if there is no requirement for the URL to be included in the categories of unwanted web resources.</p>	<ul style="list-style-type: none">• <code>drweb-dws</code>,• <code>drweb-gated</code>,• <code>drweb-firewall</code>,• <code>drweb-netcheck</code>,• <code>drweb-se</code>	<ul style="list-style-type: none">• Dr.Web ConfigD,• Dr.Web Ctl,• Dr.Web Firewall for Linux,• Dr.Web Network Checker,• Dr.Web Scanning Engine, *• Dr.Web Updater, ***• Dr.Web URL Checker,• SplDer Gate,• Database of web resource categories, **• Virus database, *• Spam filter ****
<p>* The component will not be installed if the package <code>drweb-se</code> is not installed. ** The component will not be installed if the package <code>drweb-dws</code> is not installed.</p>		



Custom Component Kit	Packages for Installation	Will be Installed
*** The Dr.Web Updater component will be installed only if virus databases, web resource category database and the spam filter are installed.		

Installation and Uninstallation of Dr.Web for UNIX Internet Gateways Components Installed from Repository

If Dr.Web for UNIX Internet Gateways 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 Dr.Web for UNIX Internet Gateways installed on CentOS, use the command:

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

2. To additionally install Dr.Web ClamD (package `drweb-clamd`) to Dr.Web for UNIX Internet Gateways installed on OS Ubuntu, use the command:

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

If needed, use help on package manager used in your OS.

Installation and Uninstallation of Dr.Web for UNIX Internet Gateways Components Installed from the Universal Package

If Dr.Web for UNIX Internet Gateways 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 Dr.Web for UNIX Internet Gateways was installed. If you did not save this file, download it from the Doctor Web website.

Unpacking the Installation File

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

`--noexec`—unpack Dr.Web for UNIX Internet Gateways 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 Dr.Web for UNIX Internet Gateways installation files and the installation log automatically after the installation completes.

`--target <directory>`—unpack Dr.Web for UNIX Internet Gateways 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 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 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 release, and `<platform>` is a platform for which Dr.Web for UNIX Internet Gateways 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 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 Dr.Web for UNIX Internet Gateways installation, launch the automatic installation script. To do that, use the command:

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

Besides that, you can install all Dr.Web for UNIX Internet Gateways packages (to install the missing or accidentally deleted components as well) by launching the installation of the root meta-package:

```
# ./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 MSVS 3.0 OS), or if there is no package manager in your system (FreeBSD), for the custom uninstallation, you should use the package manager `zypper`, which is automatically installed within the Dr.Web for UNIX Internet Gateways installation. To do that, go to the directory `<opt_dir>/bin` (for GNU/Linux—`/opt/drweb.com/bin`) and execute the command:

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

For example:

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

If you need to uninstall Dr.Web for UNIX Internet Gateways, launch the [automatic removal](#) script. To do this, enter the 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.

See below the details of [configuring SELinux security policies](#).

Configuring SELinux Security Policies

If your GNU/Linux distribution includes SELinux (*Security-Enhanced Linux*), you may need to configure SELinux security policies to get the servicing components of Dr.Web for UNIX Internet Gateways (such as the [scan engine](#)) to operate correctly after the installation.

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.

If installation of Dr.Web for UNIX Internet Gateways from the file fails due to inability to create the *drweb* user, check the SELinux operation mode with the `getenforce` command. The command outputs the current scanning mode:

- *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 command:

```
# setenforce 0
```

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



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 its settings (file with SELinux settings usually resides in the `/etc/selinux` directory).

After the successful Dr.Web for UNIX Internet Gateways installation, enable the *Enforced* mode again before starting the product. For that, use the command:

```
# setenforce 1
```

Dr.Web for UNIX Internet Gateways Operation Issues

In certain cases when SELinux is running, several Dr.Web for UNIX Internet Gateways components (such as `drweb-se` and `drweb-filecheck`) can not launch, which causes the impossibility of objects scanning and file system monitoring. The sign of the impossibility of launching these components is the appearance of [119](#) and [120](#) error messages in system log, managed by `syslog` service (typically this log is located in `/var/log/` directory).

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.



Certain GNU/Linux distributions do not feature the utilities mentioned below. If so, you may need to install additional packages with the utilities.

To Configure 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 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.



You can use this method only if Dr.Web for UNIX Internet Gateways 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 Red Hat 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.



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.



The `policygentool` utility included in the `selinux-policy` package for Red Hat Enterprise Linux and CentOS 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.



To ensure successful execution of the command, the `checkpolicy` package must be installed in the system.

Usage example:

```
# 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 your UNIX distribution.



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 Dr.Web for UNIX Internet Gateways 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. If you are not using an external proxy server, [configure](#) the proxy mode for SpIDer Gate.
6. 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).



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

In this section

- [Purchasing and Registering License](#)
- [Obtaining Demo License](#)
- [Key File Installation](#)
- [Repeated Registration](#)

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 from our [partners](#) or our [online store](#). For details on license options, visit the Doctor Web official website at https://license.drweb.com/license_manager.

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 official website at <https://products.drweb.com/register/v4>.

During the activation, it is required to enter the serial number of the purchased license. The serial number is supplied with Dr.Web for UNIX Internet Gateways or via email when purchasing or renewing the license online.

Obtaining Demo License

A demo period for your copy of Dr.Web for UNIX Internet Gateways can be obtained on the Doctor Web official website at <https://download.drweb.com/demoreq/biz/v2>. 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 GNU/Linux) under the name `drweb32.key`.

Components of Dr.Web for UNIX Internet Gateways 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 not recommended to 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 Dr.Web for UNIX Internet Gateways 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.



In email messages, Dr.Web key files are usually transferred packed in zip archives. The archive containing the key file for Dr.Web for UNIX Internet Gateways activation usually named `agent.zip`. If the message contains *several* archives, then it is necessary to use the `agent.zip` archive.

Before the key file installation, you must unpack the archive in any convenient way and extract the key file from it, saving it to any available directory (for example, to your home directory or to the 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 Dr.Web for UNIX Internet Gateways 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. Reload the setting of Dr.Web for UNIX Internet Gateways by entering the following [command](#):

```
# drweb-ctl reload
```

to apply all changes.



You can also use the [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.

Repeated 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 can use a different email address. In this case, the license key file will be sent to the newly specified address.

A license key file can be obtained through the license management command a limited number of times. If that amount has been exceeded, you can confirm the registration of your serial number at <https://products.drweb.com/register/> to receive the key file. The key file is sent to the email that was specified during the first registration.

After the key file is sent to you by email, you need to [install](#) it manually.

Testing Product Operation

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 terminal screen or to the console emulator 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 Dr.Web for UNIX Internet Gateways by means of EICAR test from the command line:

```
$ tail <opt_dir>/share/doc/drweb-se/readme.eicar | grep X50 > testfile &&  
drweb-ctl rawscan testfile && rm testfile
```

This command sets off from the file `<opt_dir>/share/doc/drweb-se/readme.eicar` (supplied with Dr.Web for UNIX Internet Gateways) a string that represents the body of the EICAR test file, then writes it into a file named `testfile` created in the current directory, then scans the resulting file and removes this file afterwards.



The above-mentioned test requires write access to the current directory. 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](#)).

To test an incoming HTTP traffic for viruses:

Via web browser

1. Open a browser and go to the proxy server settings.
2. Enter the appropriate proxy server settings for ICAP.
3. Visit the following webpage: <https://www.eicar.org/download/eicar.com>. Notification that the file is infected will appear in the browser window.



Via console

Make the following request:

```
curl -x 127.0.0.1:3128 https://www.eicar.org/download/eicar.com
```

In the response you should see the notification that the file you requested is infected.

Integration with Squid Proxy Server

In this section

- [Configuring Dr.Web ICAPD](#)
- [Configuring Squid](#)
- [Squid Advanced Settings](#)

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 the `Block*` settings, enable or disable categories of websites and threat types that Dr.Web ICAPD should block or allow.
- If required, specify the list of websites to be blocked as the value of the `BlackList` parameter. In the value of the `WhiteList` parameter you can specify the list of websites that must not be blocked.



The `BlackList` parameter takes precedence over `WhiteList`. If a domain is included in the lists for both parameters, it will be blocked.

- To configure access to websites in a more detailed 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 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 with the following [command](#):

```
# drweb-ctl reload
```

You can also restart the configuration daemon Dr.Web ConfigD with the following command:

```
# service drweb-configd restart
```

Configuring Squid

To enable the 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, proceed as follows:

1. Enable the ICAP in the settings of Squid.
2. Register Dr.Web ICAPD as the ICAP service for Squid.
3. Enable the *ICAP preview* mode (optionally).
4. Allow transferring the client data (i.e. the IP address and the user name of a user who has passed authentication at the proxy server) for using in the rules of Dr.Web ICAPD (optionally).
5. Enable the support of persistent connections between Dr.Web ICAPD and Squid (optional; though enabling persistent connections is not necessary, it can increase the performance of Squid working together with Dr.Web ICAPD).



- 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.
- The IP-address and the username of the client are sent to Dr.Web ICAPD by Squid in the the ICAP request in `X-Client-Username` and `X-Client-IP` headers. The values of these headers must encoded by the methods used in Squid by default. The settings of Squid that affect the encoding methods (`icap_client_username_encode` and `icap_client_username_header` headers) should not be modified.



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



For working with the HTTPS protocol Squid should be built with SSL support (that is, compiled with the `--with-openssl` and `--enable-ssl-crtd` options). In the settings of Squid SSL bumping should be enabled.

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).

Modify your Squid configuration file according to the given examples.

If the parameters from the examples are commented out, uncomment them. If the required parameters are missing, add them to the Squid configuration file.



Only #1 and #2 steps are obligatory for configuring the 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
```

Restart Squid after changing the settings.

Advanced Settings for Squid: data size restrictions

If necessary, you can limit the size of data that Squid sends for scanning via the ICAP protocol. To do this, specify in the configuration file the value of the `Content-Length` header (the specific size in bytes or a regular expression), for example:

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



(the condition `<name>` holds, if the header `Content-Length` in the server response contains a number greater than 999999).

The condition from the example above can be used to `allow` or `deny` the scanning of the server response via the ICAP protocol (the word `all` must be replaced in the connection parameters of Squid by the condition name `<name>`). The following example shows the settings used to `deny` the scanning of responses for which the condition `<name>` holds:

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

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



The `Content-Length` header can be missing in the webserver response. In this case the settings concerning the restrictions of data size will not be applied.

The detailed information on configuring the restrictions of web-traffic in Squid can be found [in the official documentation](#).



After editing the configuration file restart Squid in order than the modified settings take effect.

Protecting a Local Web Server

In this section

- [Configuring Redirection of Connections](#)
- [Scan Settings](#)



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

To protect a web server that is running on the same host on which Dr.Web for UNIX Internet Gateways is installed, you need enable scanning all the traffic coming to the server by the [SplDer Gate](#) monitor.

Configuring Redirection of Connections

To configure the web server protection, change several parameter values in the configuration file, in the section with the [settings](#) for Dr.Web Firewall for Linux (section `[LinuxFirewall]`):



Parameter	Required value
InspectHttp	On
AutoconfigureIptables	Yes
AutoconfigureRouting	Yes
LocalDeliveryMark	Auto
ClientPacketsMark	Auto
ServerPacketsMark	Auto
TproxyListenAddress	127.0.0.1:0 If a special IP address or port are used for the Dr.Web Firewall for Linux operation, specify them here
InputDivertEnable	Yes
InputDivertNfqueueNumber	Auto
InputDivertConnectTransparently	Yes

To view and to change the settings of Dr.Web Firewall for Linux, 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).

For example, the following command:

```
# drweb-ctl cfset LinuxFirewall.InputDivertEnable Yes
```

will configure Dr.Web Firewall for Linux so that the the incoming data be scanned by SpiDer Gate if the HTTP protocol is used and the `InspectHttp` parameter value is set to `On`.

- 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/`).

To scan data transferred via HTTPS protocol, additionally do the following:

- Enable scanning of the traffic transmitted via SSL/TLS by indicating the value of the corresponding parameter by executing the command:

```
# drweb-ctl cfset LinuxFirewall.UnwrapSsl Yes
```

It is recommended that the command `cfset` of the tool `drweb-ctl` or management web interface is used, because in this case the scanning rules will change automatically. They depend on this parameter.



- Export a certificate, which will be used by Dr.Web for UNIX Internet Gateways for integration into the protected SSL/TLS channels by executing the command :

```
$ drweb-ctl certificate > <cert_name>.pem
```

It is necessary to indicate the name of the file used for saving the certificate in the PEM format.

- Add an obtained certificate to the system list of trusted certificates and, possible, write it as the trusted certificate for web clients (browsers) and the web server. For details, see [Appendix E. Generating SSL certificates](#) section.

Scan Settings

Specify the following parameters in the `LinuxFirewall` section of the configuration file:

1. Parameters of scanning of transferred data (`ScanTimeout`, `HeuristicAnalysis`, `PackerMaxLevel`, `ArchiveMaxLevel`, `MailMaxLevel`, `ContainerMaxLevel`, `MaxCompressionRatio`) that limit the length and resource intensity of their scanning. When a fine-grained configuration is not required, it is recommended that values for parameter data are kept in their default state.
2. The `Block*` parameters for blocking unwanted URLs and content.
3. The `BlockUnchecked` parameter to specify the actions of the SpIDer Gate in case the received data cannot be scanned.

For a more detailed configuration of filtering rules edit the [Lua procedure](#) or the [RuleSet rules](#).

After all settings are adjusted, restart Dr.Web for UNIX Internet Gateways with the following [command](#):

```
# drweb-ctl reload
```

You can also restart the configuration daemon Dr.Web ConfigD with the following command:

```
# service drweb-configd restart
```

Using SpIDer Gate in Proxy Mode

In this section

- [Configuring the Proxy Mode](#)
- [Scan settings](#)



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



To protect a local network from threats spread via the internet, if the HTTP proxy server, which could [communicate](#) with Dr.Web for UNIX Internet Gateways via ICAP or [over](#) the ClamAV protocol (using the [Dr.Web ClamD](#) component directly), is missing on the internet gateway, configure the [Dr.Web Firewall for Linux](#) so that information received via the internet gateway, with Dr.Web for UNIX Internet Gateways installed on it, were scanned by the [SpIDer Gate](#) monitor (a transparent proxy mode).

Configuring the Proxy Mode

To configure the Transparent Proxy Mode, change several parameter values in the configuration file, in the section with the [settings](#) for Dr.Web Firewall for Linux (section [LinuxFirewall]):

Parameter	Required value
InspectHttp	On
AutoconfigureIptables	Yes
AutoconfigureRouting	Yes
LocalDeliveryMark	Auto
ClientPacketsMark	Auto
ServerPacketsMark	Auto
TproxyListenAddress	127.0.0.1:0 If a special IP address or port are used for the Dr.Web Firewall for Linux operation, specify them here
ForwardDivertEnable	Yes
FowardDivertNfqueueNumber	Auto
ForwardDivertConnectTransparently	Yes

To view and to change the settings of Dr.Web Firewall for Linux, 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).

For example, the following command:

```
# drweb-ctl cfset LinuxFirewall.ForwardDivertEnable Yes
```

will configure Dr.Web Firewall for Linux in the following way. The incoming data will be scanned by SpIDer Gate if the HTTP protocol is used and the corresponding `InspectHttp` parameter value is set to `On`.



- 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/`).

To scan data transferred via HTTPS protocol:

- Enable the scanning of the traffic transmitted via SSL/TLS:

```
# drweb-ctl cfset LinuxFirewall.UnwrapSsl Yes
```

Use the `cfset` command of the tool `drweb-ctl` or the web interface to apply a new value for this parameter so that the values of all dependent parameters be changed automatically.

- Export the certificate that will be used by Dr.Web for UNIX Internet Gateways for integration into the protected SSL/TLS channels by executing the command:

```
$ drweb-ctl certificate > <cert_name>.pem
```

It is necessary to indicate the name of the file used for saving the certificate in the PEM format.

- Add the certificate to the system list of trusted certificates and specify it as the trusted certificate for web clients (browsers) and the web server. For details, see [Appendix E. Generating SSL certificates](#) section.

Scan Settings

Specify the following parameters in the `LinuxFirewall` section of the configuration file:

1. Parameters of scanning of transferred data (`ScanTimeout`, `HeuristicAnalysis`, `PackerMaxLevel`, `ArchiveMaxLevel`, `MailMaxLevel`, `ContainerMaxLevel`, `MaxCompressionRatio`) that limit the length and resource intensity of their scanning. When a fine-grained configuration is not required, it is recommended that values for parameter data are kept in their default state.
2. The `Block*` parameters for blocking unwanted URLs and content.
3. The `BlockUnchecked` parameter to specify the actions of the SpIDer Gate in case the received data cannot be scanned.

For a more detailed configuration of filtering rules edit the [Lua procedure](#) or the `RuleSet rules`.

After all settings are adjusted, restart Dr.Web for UNIX Internet Gateways with the following [command](#):

```
# drweb-ctl reload
```

You can also restart the configuration daemon Dr.Web ConfigD with the following command:

```
# service drweb-configd restart
```



Brief Instructions

In this section

- Working with HTTP Proxies and Web Servers:
 - [How to Connect Dr.Web for UNIX Internet Gateways to Squid](#)
 - [How to Protect a Web Server](#)
 - [How to Configure Proxy Mode for SplDer Gate](#)
- General Operation of Dr.Web for UNIX Internet Gateways:
 - [How to Restart Dr.Web for UNIX Internet Gateways](#)
 - [How to Connect to the centralized protection server](#)
 - [How to Disconnect From the Centralized Protection Server](#)
 - [How to Activate Dr.Web for UNIX Internet Gateways](#)
 - [How to Upgrade Dr.Web for UNIX Internet Gateways](#)
 - [How To Add or Remove Dr.Web for UNIX Internet Gateways Component](#)
 - [How to Manage Dr.Web for UNIX Internet Gateways Component Operation](#)
 - [How to View Log of the Dr.Web for UNIX Internet Gateways](#)

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 Configure Proxy Mode for SplDer Gate

Follow the instructions provided in the [Using SplDer Gate in Proxy Mode](#) section.



How to Restart Dr.Web for UNIX Internet Gateways

To restart Dr.Web for UNIX Internet Gateways 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 shell script that controls the operation of Dr.Web ConfigD is residing in the standard OS directory (for GNU/Linux—`/etc/init.d/`; for FreeBSD—`/usr/local/etc/rc.d/`). The name of the script is `drweb-configd`. It has the following parameters:

Parameter	Description
<code>start</code>	Start Dr.Web ConfigD if it is not running. When Dr.Web ConfigD starts, Dr.Web ConfigD launches all the required components of Dr.Web for UNIX Internet Gateways.
<code>stop</code>	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.
<code>restart</code>	Restart (shut down and then start) Dr.Web ConfigD. Dr.Web ConfigD shuts down and then starts all the components of Dr.Web for UNIX Internet Gateways. If Dr.Web ConfigD is not running, the parameter has the same effect as <code>start</code> .
<code>condrestart</code>	Restart Dr.Web ConfigD only if it is running.
<code>reload</code>	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.
<code>status</code>	Output the current state of Dr.Web ConfigD to the console.

For example, to restart Dr.Web for UNIX Internet Gateways (or start it, if it is not running) in GNU/Linux OS, use the following command:

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

How to Connect to the Centralized Protection Server

1. Obtain the address of the centralized protection server and the file of its certificate 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 with Dr.Web for UNIX Internet Gateways.

For connection it is required to use the option `--Certificate` by specifying the path to the certificate 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 centralized protection server) and a password for authentication on the centralized 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 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.



In this mode the centralized 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 centralized protection server:

```
# drweb-ctl esconnect <server address> --Certificate <path to the certificate file>
```

After establishing a connection to the centralized protection server, Dr.Web for UNIX Internet Gateways will operate in the centralized protection mode or in the mobile mode, depending on the permissions set on the server and the value of the `MobileMode` [configuration parameter](#) of the Dr.Web ES Agent component. To allow unconditional use of the mobile mode, set the parameter value to `On`. For operation in the centralized protection mode, set the parameter value to `Off`.

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

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



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

How to Disconnect From the Centralized Protection Server

To disconnect Dr.Web for UNIX Internet Gateways from the centralized 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 Dr.Web for UNIX Internet Gateways in standalone mode, a valid license [key file](#) is required. Otherwise, anti-virus functions of Dr.Web for UNIX Internet Gateways will be *blocked* after the operation is switched to standalone mode.

How to Activate Dr.Web for UNIX Internet Gateways

1. Register on Doctor Web website at <https://products.drweb.com/register/v4>.
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 Dr.Web for UNIX Internet Gateways

[Update](#) component versions or [upgrade to a new version](#).



During the upgrade you can be asked to remove the current Dr.Web for UNIX Internet Gateways version.

How To Add or Remove Dr.Web for UNIX Internet Gateways Component

Follow the [Custom Component Installation and Uninstallation](#) procedure.



When installing and uninstalling the component, other Dr.Web for UNIX Internet Gateways components could be additionally installed or uninstalled to resolve dependencies.

How to Manage Components Operation

To view the status of Dr.Web for UNIX Internet Gateways 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 Dr.Web for UNIX Internet Gateways

According to default settings the general log of all Dr.Web for UNIX Internet Gateways 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.

Also you can use the `drweb-ctl log` [command](#).

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
```



Dr.Web for UNIX Internet Gateways Components

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

Dr.Web ConfigD

Dr.Web ConfigD configuration management daemon is the main control component of Dr.Web for UNIX Internet Gateways. It provides the storage of the configurations of all Dr.Web for UNIX Internet Gateways components in the centralized environment, manages the operation of all components, and organizes trusted data exchange between them.

The Dr.Web ConfigD configuration management daemon performs the following functions:

- starting and stopping the components of the Dr.Web for UNIX Internet Gateways depending on settings;
- restarting the components automatically in case of failures;
- starting components upon the request of other components;
- informing the components upon the modification of the settings;
- providing the interface for the centralized management of configuration parameters;
- conveying the information from the license file in use to the components;
- accepting the license information from the components;
- receiving the new license information from the specialized components;
- informing the running components upon the modification of the license information.

Operating Principles

The Dr.Web ConfigD component is always run as root. It starts the other Dr.Web for UNIX Internet Gateways components and interacts with the them via the preliminarily open socket. The configuration management daemon can accept connections from the other components of the Dr.Web for UNIX Internet Gateways via the information socket (accessible to all components) and via the managing socket (accessible only to the components run as root). It loads the configuration and the license information from files or from the centralized protection server via the [Dr.Web ES Agent](#). It sets the correct default values for the configuration parameters. By the time when any component starts or receives a `SIGHUP` signal Dr.Web ConfigD has a comprehensive and consistent set of configuration parameters for all Dr.Web for UNIX Internet Gateways components.

Upon the receipt of a `SIGHUP` signal, Dr.Web ConfigD re-reads the configuration parameters and the license information. If required, the daemon sends to all components notifications instructing them to re-read their configuration.



Upon the receipt of a `SIGTERM` signal, Dr.Web ConfigD shuts down all components and then finishes its own operation. Dr.Web ConfigD also removes all temporary files of the components after they are shut down.

Component Interaction Principles

1. At startup all components receive the configuration parameters and the license information from Dr.Web ConfigD. Only these parameters are used in further operation.
2. The daemon collects messages from all the controlled components into an integrated log. All information output to `stderr` component is collected by Dr.Web ConfigD and written to the integrated log of Dr.Web for UNIX Internet Gateways 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 Dr.Web for UNIX Internet Gateways log.
 - [Code 101](#) is returned when the component cannot operate with the current license. The component will be restarted only after the modification of license parameters.
 - [Code 102](#) is returned when the component cannot operate with the current configuration parameters. Dr.Web ConfigD will try to restart the component when the configuration parameters are modified.
 - Code 103 is returned in case the components started by Dr.Web ConfigD upon request ([Dr.Web Scanning Engine](#) and [Dr.Web File Checker](#)) have been idle for a long time. The period of time after which the component is to shut down is specified in its settings (`IdleTimeLimit` parameter).
 - If the configuration parameters received from the configuration management daemon cannot be applied on the fly, the component exists with code 0 so that Dr.Web ConfigD restart it.
 - 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 is organized.
 - Dr.Web ConfigD sends the `SIGHUP` signal to components in order that they apply the modified configuration parameters.
 - Dr.Web ConfigD sends the `SIGTERM` signal to the components so that they shut down. After receiving the signal the component is to shut down in 30 seconds.
 - If a component does not shut down in 30 seconds, Dr.Web ConfigD sends the `SIGKILL` signal to shut it down forcibly.



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
<code>--help</code>	Function: Output short help information about command-line parameters to the console or to the terminal emulator and to exit upon completion. Short form: <code>-h</code> Arguments: None.
<code>--version</code>	Function: Output information about the version of this component to the console or to the terminal emulator and to exit after completion. Short form: <code>-v</code> Arguments: None.
<code>--config</code>	Description: Use the specified configuration file for further operation. Short form: <code>-c</code> Arguments: <i><path to the file></i> —the path to the configuration file that you want to use.
<code>--daemonize</code>	Description: Run the component in daemon mode. Short form: <code>-d</code> Arguments: None.
<code>--pid-file</code>	Description: Use the specified PID file for further operation. Short form: <code>-p</code> Arguments: <i><path to the file></i> —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 Dr.Web for UNIX Internet Gateways, Dr.Web ConfigD must run as a daemon. During the standard booting, Dr.Web ConfigD starts automatically at the start of the operating system; for this purpose Dr.Web ConfigD comes together with a standard management script `drweb-configd` located in the standard OS directory (for GNU/Linux—`/etc/init.d/`; for FreeBSD—`/usr/local/etc/rc.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 the configuration parameters which can be found in the [Root] section of the integrated [configuration file](#) of Dr.Web for UNIX Internet Gateways.

The section contains the following parameters:

Parameter	Description
<code>DefaultLogLevel</code> <i>{logging level}</i>	Defines default logging level of event logging <i>for all</i> the Dr.Web for UNIX Internet Gateways components. The value of this parameter is used for all the components in the product for which the logging level cannot be set up separately. Default value: <code>Notice</code>
<code>LogLevel</code> <i>{logging level}</i>	Logging level of event logging for Dr.Web ConfigD. Default value: <code>Notice</code>
<code>Log</code> <i>{log type}</i>	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: <ul style="list-style-type: none">• As a daemon (if run with the <code>-d</code> option)—<code>SYSLOG:Daemon</code>• Otherwise—<code>Stderr</code> If a component is working in a background mode (was launched with the <code>-d</code> option from the command line), the <code>Stderr</code> value <i>cannot be used</i> for this parameter. Default value: <code>SYSLOG:Daemon</code>
<code>PublicSocketPath</code> <i>{path to file}</i>	Path to the socket used for interaction between all the Dr.Web for UNIX Internet Gateways components. Default value: <code>/var/run/.com.drweb.public</code>
<code>AdminSocketPath</code> <i>{path to file}</i>	Path to the socket used for interaction between the Dr.Web for UNIX Internet Gateways components with elevated (administrative) privileges. Default value: <code>/var/run/.com.drweb.admin</code>



Parameter	Description
CoreEnginePath <i>{path to file}</i>	<p>Path to the dynamic library of the Dr.Web Virus-Finding Engine scan engine.</p> <p>Default value: <code><var_dir>/lib/drweb32.dll</code></p> <ul style="list-style-type: none">• For GNU/Linux: <code>/var/opt/drweb.com/lib/drweb32.dll</code>• For FreeBSD: <code>/var/drweb.com/lib/drweb32.dll</code>
VirusBaseDir <i>{path to directory}</i>	<p>Path to the directory with virus database files.</p> <p>Default value: <code><var_dir>/bases</code></p> <ul style="list-style-type: none">• For GNU/Linux: <code>/var/opt/drweb.com/bases</code>• For FreeBSD: <code>/var/drweb.com/bases</code>
KeyPath <i>{path to file}</i>	<p>Path to the key file (license or demo).</p> <p>Default value: <code><etc_dir>/drweb32.key</code></p> <ul style="list-style-type: none">• For GNU/Linux: <code>/etc/opt/drweb.com/drweb32.key</code>• For FreeBSD: <code>/usr/local/etc/drweb.com/drweb32.key</code>
CacheDir <i>{path to directory}</i>	<p>Path to the cache directory (used to hold cache for updates as well as cache for information about scanned files).</p> <p>Default value: <code><var_dir>/cache</code></p> <ul style="list-style-type: none">• For GNU/Linux: <code>/var/opt/drweb.com/cache</code>• For FreeBSD: <code>/var/drweb.com/cache</code>
TempDir <i>{path to directory}</i>	<p>Path to the directory with temporary files.</p> <p>Default value: <i>Path copied from the system environment variable</i> <code>TMPDIR</code>, <code>TMP</code>, <code>TEMP</code> or <code>TEMPDIR</code> (the environment variables are searched in this particular order). Otherwise <code>/tmp</code>, if there are no such environment variables.</p>
RunDir <i>{path to directory}</i>	<p>Path to the directory with all PID files of running components and sockets used for interaction between the Dr.Web for UNIX Internet Gateways components.</p> <p>Default value: <code>/var/run</code></p>
VarLibDir <i>{path to directory}</i>	<p>Path to the directory with libraries used by Dr.Web for UNIX Internet Gateways components.</p> <p>Default value: <code><var_dir>/lib</code></p> <ul style="list-style-type: none">• For GNU/Linux: <code>/var/opt/drweb.com/lib</code>• For FreeBSD: <code>/var/drweb.com/lib</code>
VersionDir <i>{path to directory}</i>	<p>The path to a directory, where the information on the Dr.Web for UNIX Internet Gateways components current versions is stored.</p> <p>Default value: <code><var_dir>/version</code></p> <ul style="list-style-type: none">• For GNU/Linux: <code>/var/opt/drweb.com/version</code>



Parameter	Description
	<ul style="list-style-type: none">• For FreeBSD: <code>/var/drweb.com/version</code>
DwsDir <i>{path to directory}</i>	<p>Path to the directory that contains files of an automatically updated database of internet resource categories.</p> <p>Default value: <code><var_dir>/dws</code></p> <ul style="list-style-type: none">• For GNU/Linux: <code>/var/opt/drweb.com/dws</code>• For FreeBSD: <code>/var/drweb.com/dws</code>
AdminGroup <i>{group name GID}</i>	<p>Group of users with administrative privileges for Dr.Web for UNIX Internet Gateways management. These users, in addition to the <code>root</code> superuser, are allowed to elevate privileges of the Dr.Web for UNIX Internet Gateways components to superuser privileges.</p> <p>Default value: <i>Determined</i> during the Dr.Web for UNIX Internet Gateways installation.</p>
TrustedGroup <i>{group name GID}</i>	<p>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.</p> <p>Note that you cannot specify a non-existent group here, as in this case SpIDer Gate will fail to start.</p> <p>If the parameter value is missing, you cannot specify the <code>Auto</code> value for the <code>OutputDivert</code> parameter in SpIDer Gate settings.</p> <p>Default value: <code>drweb</code></p>
DebugIpc <i>{Boolean}</i>	<p>Include detailed into the log file on the debug level (i.e. when <code>LogLevel = DEBUG</code>). IPC messages show the interaction between the configuration daemon and other components.</p> <p>Default value: <code>No</code></p>
UseCloud <i>{Boolean}</i>	<p>Use/do not use the Dr.Web Cloud service to receive information about malicious files and URLs.</p> <p>Default value: <code>No</code></p>
AntispamCorePath <i>{path to file}</i>	<p>The parameter is not used.</p> <p>Default value: <code><var_dir>/lib/vaderetro.so</code></p> <ul style="list-style-type: none">• For GNU/Linux: <code>/var/opt/drweb.com/lib/vaderetro.so</code>• For FreeBSD: <code>/var/drweb.com/lib/vaderetro.so</code>
VersionNotification <i>{Boolean}</i>	<p>Notify a user on availability of updates for the currently installed Dr.Web for UNIX Internet Gateways version.</p> <p>Default value: <code>Yes</code></p>
UseVxcube <i>{Boolean}</i>	<p>Use Dr.Web vxCube for analyzing email attachments when in the mode of an external filter connected to the MTA.</p> <p>Default value: <code>No</code></p>



Parameter	Description
VxcubeApiAddress {string}	The domain name (FQDN) or the IP address of the host on which the Dr.Web vxCube API server is running. Default value: <i>(not set)</i>
VxcubeApiKey {string}	Dr.Web vxCube API key. Default value: <i>(not set)</i>
VxcubeProxyUrl {connection address}	The address of the proxy server used for connecting to Dr.Web vxCube. Only HTTP proxies without authorization are supported. Possible values: <connection address>—proxy server connection parameters in the following format: <code>http://<host>:<port></code> , where: <ul style="list-style-type: none">• <host> is the host address of the proxy server (IP address or domain name, i.e. FQDN);• <port> is the port to be used. Default value: <i>(not set)</i>

Dr.Web Ctl

In this section

- [General Information](#)
- [Remote host scanning](#)

General Information

You can manage operation of Dr.Web for UNIX Internet Gateways from the command line of the operating system. For that, you can use the special Dr.Web Ctl utility (`drweb-ctl`). You can use it to perform the following operations:

- 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, the scan engine, and so on depending on the distribution).
- View and change parameters of the Dr.Web for UNIX Internet Gateways configuration.
- View the status of the Dr.Web for UNIX Internet Gateways components and statistics on detected threats.
- Connect to the centralized protection server or disconnect from it.
- View quarantine and manage quarantined objects (via the [Dr.Web File Checker](#) component).
- Connect to the centralized protection server or disconnect from it.



User [commands](#) to control Dr.Web for UNIX Internet Gateways will only take effect if the [Dr.Web ConfigD](#) configuration daemon is running (by default, it is automatically run on system 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 `drweb-ctl` tool supports auto-completion of commands for managing Dr.Web for UNIX Internet Gateways operation if this option is enabled your 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 only returns a non-null exit code in the case of internal error (for example, the tool could not connect to a component, the requested operation could not be executed, and so on). If the tool detects and possibly neutralizes a threat, it returns the null exit code, because the requested operation (such as `scan`, and so on) is successfully completed. If you need to define the list of the 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 you to scan files located on remote network hosts for threats. Such hosts can be not only fully-featured computing machines, such as workstations and servers, but also routers, set-top boxes, and other smart devices of the Internet of Things. To perform the remote scanning, the remote host has to provide a remote terminal access via *SSH (Secure Shell)* or *Telnet*. To access the device, you need to know an IP address and a domain name of the remote host, as well as the credentials of the user that can remotely access the system via *SSH* or *Telnet*. This 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 cure of malicious objects) using remote scanning is impossible. To eliminate the detected threats on the remote host, use administration tools provided directly by this host. For example, for routers and other smart devices, update the firmware; for computing machines, establish a connection (in a remote terminal mode, as one of the options) and perform the respective operations in the file system (remove or move files, etc.), or run the anti-virus software installed on them.



Remote scanning is only performed via the command-line tool `drweb-ctl` (using the [command](#) `remotescan`).

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



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 the centralized protection mode.
- [Configuration management](#) commands.
- Commands to [manage detected threats and quarantine](#).
- [Information](#) commands.



To request help 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
<code>scan <path></code>	<p>Purpose: to initiate the scan of the specified file or directory by the file scanning component Dr.Web File Checker.</p> <p>Arguments</p> <p><code><path></code>—path to the file or directory to be scanned (the path can be relative).</p> <p>This argument may be omitted if you use the <code>--stdin</code> or the <code>--stdin0</code> option. To specify several files that satisfy a certain criterion, use the <code>find</code> utility (see Usage Examples) and the <code>--stdin</code> or <code>--stdin0</code> option.</p> <p>Options</p> <p><code>-a [--Autonomous]</code> runs an autonomous copy of Dr.Web Scanning Engine and Dr.Web File Checker to perform the specified scan, terminating them after it is over. Note that threats detected during autonomous scanning will not be added to the common list of threats detected displayed by <code>threats</code> command (see below), and information on them will not be sent to the centralized protection server, if Dr.Web for UNIX Internet Gateways is controlled by it.</p> <p><code>--stdin</code>—get the list of paths to scan from the standard input string (<code>stdin</code>). Paths in the list need to be separated by the next line character (<code>'\n'</code>).</p> <p><code>--stdin0</code>—get the list of paths to scan from the standard input string (<code>stdin</code>). Paths in the list need to be separated by the zero character NUL (<code>'\0'</code>).</p>



Command	Description
	<div data-bbox="608 259 1449 521" style="background-color: #e6f2e6; padding: 10px;"> When using <code>--stdin</code> and <code>--stdin0</code> options, the paths on the list should not contain patterns or regular expressions for a search. We recommend that you use the <code>--stdin</code> and <code>--stdin0</code> options to process a paths list generated by an external utility, for example, <code>find</code> in the <code>scan</code> command (see Usage Examples).</div> <p><code>--Exclude <path></code>—an excluded path. The path can be relative and contains a file mask (with the following wildcards: '?', '*', as well as symbol classes '[]', '[!]', '[^]').</p> <p>Facultative option; can be set more than once.</p> <p><code>--Report <type></code>—specifies the type of scan report.</p> <p>Allowed values:</p> <ul style="list-style-type: none">• BRIEF—brief report.• DEBUG—detailed report.• JSON—a serialized report in JSON format. <p>Default value: BRIEF.</p> <p><code>--ScanTimeout <number></code>—specify time-out to scan one file, in ms.</p> <p>If the value is set to 0, time on scanning is not limited.</p> <p>Default value: 0.</p> <p><code>--PackerMaxLevel <number></code>—set the maximum nesting level when scanning packed objects. A packed object is executable code compressed with special software (UPX, PELock, PECompact, Petite, ASPack, Morphine and so on). Such objects may include other packed objects which may also include packed objects. etc. The value of this parameter specifies the nesting limit beyond which packed objects inside other packed objects will not be scanned.</p> <p>If the value is set to 0, nested objects will be skipped during scanning.</p> <p>Default value: 8.</p> <p><code>--ArchiveMaxLevel <number></code>—set the maximum nesting level when scanning archives (zip, rar, and so on) in which other archives may be enclosed (and these archives may also include other archives, and so on). The value of this parameter specifies the nesting limit beyond which archives enclosed in other archives will not be scanned.</p> <p>If the value is set to 0, nested objects will be skipped during scanning.</p> <p>Default value: 8.</p> <p><code>--MailMaxLevel <number></code>—set the maximum nesting level when scanning files of mailers (pst, tbb and so on) in which other files may be enclosed (and these files may also include other files and so on). The value of this parameter specifies the nesting limit beyond which objects inside other objects will not be scanned.</p> <p>If the value is set to 0, nested objects will be skipped during scanning.</p>



Command	Description
	<p>Default value: 8.</p> <p><code>--ContainerMaxLevel <number></code>—set the maximum nesting level when scanning other types objects inside which other objects are enclosed (HTML pages, jar-files, etc.). The value of this parameter specifies the nesting limit beyond which objects inside other objects will not be scanned.</p> <p>If the value is set to 0, nested objects will be skipped during scanning.</p> <p>Default value: 8.</p> <p><code>--MaxCompressionRatio <ratio></code>—set the maximum compression ratio of scanned objects.</p> <p>The ratio must be at least equal to 2.</p> <p>Default value: 3000.</p> <p><code>--MaxSizeToExtract <size></code>—specify the maximum size for files enclosed in archives. Files whose size is greater than the value of this parameter will be skipped when scanning. There is no size limit for files in archives by default. The size is specified as a number with a suffix (b, kb, mb, gb). If no suffix is specified, the value is treated as size in bytes.</p> <p>Default value: none.</p> <p><code>--HeuristicAnalysis <On Off></code>—enable or disable heuristic analysis during the scanning.</p> <p>Default value: On.</p> <p><code>--OnKnownVirus <action></code>—an action to perform upon detection of a known threat by using signature-based analysis.</p> <p>Possible actions: Report, Cure, Quarantine, Delete.</p> <p>Default value: Report.</p> <p><code>--OnIncurable <action></code>—an action to perform upon detection an incurable threat or when curing action (Cure) failed.</p> <p>Possible actions: Report, Quarantine, Delete.</p> <p>Default value: Report.</p> <p><code>--OnSuspicious <action></code>—an action to perform upon detection of a suspicious object by heuristic analysis.</p> <p>Possible actions: Report, Quarantine, Delete.</p> <p>Default value: Report.</p> <p><code>--OnAdware <action></code>—an action to perform upon detection of adware programs.</p> <p>Possible actions: Report, Quarantine, Delete.</p> <p>Default value: Report.</p> <p><code>--OnDialers <action></code>—an action to perform upon detection of a dialer.</p> <p>Possible actions: Report, Quarantine, Delete.</p> <p>Default value: Report.</p>



Command	Description
	<p><code>--OnJokes <action></code>—an action to perform upon detection of joke software.</p> <p>Possible actions: Report, Quarantine, Delete.</p> <p>Default value: Report.</p> <p><code>--OnRiskware <action></code>—an action to perform upon detection of riskware.</p> <p>Possible actions: Report, Quarantine, Delete.</p> <p>Default value: Report.</p> <p><code>--OnHacktools <action></code>—an action to perform upon detection of a hacktool.</p> <p>Possible actions: Report, Quarantine, Delete.</p> <p>Default value: Report.</p> <div data-bbox="608 792 1449 981" style="background-color: #e6f2e6; padding: 10px;"> If threat is detected in a file placed in a container (an archive, an email message, and so on), instead of removing the file (Delete), the tool moves the container to the quarantine (Quarantine).</div> <p><code>--FollowSymlinks</code>—resolve symlinks automatically</p>
<code>bootscan</code> <code><disk drive></code> ALL	<p>Purpose: start scanning boot records on the specified disks via the file scan component Dr.Web File Checker. Both MBR and VBR records are scanned.</p> <p>Arguments</p> <p><code><disk drive></code>—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.</p> <p>Options</p> <p><code>-a [--Autonomous]</code> runs an autonomous copy of Dr.Web Scanning Engine and Dr.Web File Checker to perform the specified scan, terminating them after it is over. Note that threats detected during autonomous scanning will not be added to the common list of detected threats displayed by <code>threats</code> command (see below), and information on them will not be sent to the centralized protection server, if Dr.Web for UNIX Internet Gateways is controlled by it.</p> <p><code>--Report <type></code>—specifies the type of scan report.</p> <p>Allowed values:</p> <ul style="list-style-type: none">• BRIEF—brief report.• DEBUG—detailed report.• JSON—a serialized report in JSON format. <p>Default value: BRIEF.</p>



Command	Description
	<p><code>--ScanTimeout <number></code>—specify time-out to scan one file, in ms. If the value is set to 0, time on scanning is not limited. Default value: 0.</p> <p><code>--HeuristicAnalysis <On Off></code>—enable or disable heuristic analysis during the scanning. Default value: On.</p> <p><code>--Cure <Yes No></code>—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.</p> <p><code>--ShellTrace</code>—enable display of additional debug information when scanning a boot record</p>
procscan	<p>Purpose: initiates scanning of executables containing the code of currently running system processes with the Dr.Web File Checker component. If a malicious executable file is detected, it is neutralized, and all processes run by this file are forced to terminate.</p> <p>Arguments: none.</p> <p>Options</p> <p><code>-a [--Autonomous]</code> runs an autonomous copy of Dr.Web Scanning Engine and Dr.Web File Checker to perform the specified scan, terminating them after it is over. Note that threats detected during autonomous scanning will not be added to the common list of detected threats displayed by <code>threats</code> command (see below), and information on them will not be sent to the centralized protection server, if Dr.Web for UNIX Internet Gateways is controlled by it.</p> <p><code>--Report <type></code>—specifies the type of scan report.</p> <p>Allowed values:</p> <ul style="list-style-type: none">• BRIEF—brief report.• DEBUG—detailed report.• JSON—a serialized report in JSON format. <p>Default value: BRIEF.</p> <p><code>--ScanTimeout <number></code>—specify time-out to scan one file, in ms. If the value is set to 0, time on scanning is not limited. Default value: 0.</p> <p><code>--HeuristicAnalysis <On Off></code>—enable or disable heuristic analysis during the scanning. Default value: On.</p> <p><code>--PackerMaxLevel <number></code>—set the maximum nesting level when scanning packed objects. A packed object is executable code compressed with special software (UPX, PELock, PECompact, Petite, ASPack, Morphine</p>



Command	Description
	<p>and so on). Such objects may include other packed objects which may also include packed objects. etc. The value of this parameter specifies the nesting limit beyond which packed objects inside other packed objects will not be scanned.</p> <p>If the value is set to 0, nested objects will be skipped during scanning.</p> <p>Default value: 8.</p> <p><code>--OnKnownVirus <action></code>—an action to perform upon detection of a known threat by using signature-based analysis.</p> <p>Possible actions: Report, Cure, Quarantine, Delete.</p> <p>Default value: Report.</p> <p><code>--OnIncurable <action></code>—an action to perform upon detection an incurable threat or when curing action (Cure) failed.</p> <p>Possible actions: Report, Quarantine, Delete.</p> <p>Default value: Report.</p> <p><code>--OnSuspicious <action></code>—an action to perform upon detection of a suspicious object by heuristic analysis.</p> <p>Possible actions: Report, Quarantine, Delete.</p> <p>Default value: Report.</p> <p><code>--OnAdware <action></code>—an action to perform upon detection of adware programs.</p> <p>Possible actions: Report, Quarantine, Delete.</p> <p>Default value: Report.</p> <p><code>--OnDialers <action></code>—an action to perform upon detection of a dialer.</p> <p>Possible actions: Report, Quarantine, Delete.</p> <p>Default value: Report.</p> <p><code>--OnJokes <action></code>—an action to perform upon detection of joke software.</p> <p>Possible actions: Report, Quarantine, Delete.</p> <p>Default value: Report.</p> <p><code>--OnRiskware <action></code>—an action to perform upon detection of riskware.</p> <p>Possible actions: Report, Quarantine, Delete.</p> <p>Default value: Report.</p> <p><code>--OnHacktools <action></code>—an action to perform upon detection of a hacktool.</p> <p>Possible actions: Report, Quarantine, Delete.</p> <p>Default value: Report.</p>



Command	Description
	<div data-bbox="608 259 1449 412" style="background-color: #e6f2e6; padding: 10px;"> Note that if a threat is detected in an executable file, Dr.Web for UNIX Internet Gateways terminates all processes started by the file.</div>
netscan [<i><path></i>]	<p>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 scan engine (similar to the <code>scan</code> command).</p> <p>Arguments</p> <p><i><path></i>—path to the file or directory which is selected to be scanned.</p> <p>If this argument is omitted, the data received via the input thread <code>stdin</code> is scanned.</p> <p>Options</p> <p><code>--Report <type></code>—specifies the type of scan report.</p> <p>Allowed values:</p> <ul style="list-style-type: none">• BRIEF—brief report.• DEBUG—detailed report.• JSON—a serialized report in JSON format. <p>Default value: BRIEF.</p> <p><code>--ScanTimeout <number></code>—specify time-out to scan one file, in ms.</p> <p>If the value is set to 0, time on scanning is not limited.</p> <p>Default value: 0.</p> <p><code>--HeuristicAnalysis <On Off></code>—enable or disable heuristic analysis during the scanning.</p> <p>Default value: On.</p> <p><code>--PackerMaxLevel <number></code>—set the maximum nesting level when scanning packed objects. A packed object is executable code compressed with special software (UPX, PELock, PECompact, Petite, ASPack, Morphine and so on). Such objects may include other packed objects which may also include packed objects. etc. The value of this parameter specifies the nesting limit beyond which packed objects inside other packed objects will not be scanned.</p> <p>If the value is set to 0, nested objects will be skipped during scanning.</p> <p>Default value: 8.</p> <p><code>--ArchiveMaxLevel <number></code>—set the maximum nesting level when scanning archives (zip, rar, and so on) in which other archives may be enclosed (and these archives may also include other archives, and so on). The value of this parameter specifies the nesting limit beyond which archives enclosed in other archives will not be scanned.</p>



Command	Description
	<p>If the value is set to 0, nested objects will be skipped during scanning. Default value: 8.</p> <p><code>--MailMaxLevel <number></code>—set the maximum nesting level when scanning files of mailers (pst, tbb and so on) in which other files may be enclosed (and these files may also include other files and so on). The value of this parameter specifies the nesting limit beyond which objects inside other objects will not be scanned.</p> <p>If the value is set to 0, nested objects will be skipped during scanning. Default value: 8.</p> <p><code>--ContainerMaxLevel <number></code>—set the maximum nesting level when scanning other types objects inside which other objects are enclosed (HTML pages, jar-files, etc.). The value of this parameter specifies the nesting limit beyond which objects inside other objects will not be scanned.</p> <p>If the value is set to 0, nested objects will be skipped during scanning. Default value: 8.</p> <p><code>--MaxCompressionRatio <ratio></code>—set the maximum compression ratio of scanned objects.</p> <p>The ratio must be at least equal to 2. Default value: 3000.</p> <p><code>--MaxSizeToExtract <size></code>—specify the maximum size for files enclosed in archives. Files whose size is greater than the value of this parameter will be skipped when scanning. There is no size limit for files in archives by default. The size is specified as a number with a suffix (b, kb, mb, gb). If no suffix is specified, the value is treated as size in bytes. Default value: none.</p> <p><code>--Cure <Yes/No></code>—enable or disable attempts to cure detected threats.</p> <p>If the value is set to No, only a notification about a detected threat is displayed. Default value: No</p>
<code>flowscan <path></code>	<p>Purpose: start scanning the specified file or directory via Dr.Web File Checker using the “flow” method.</p> <div data-bbox="608 1563 1449 1686" style="background-color: #fff9c4; padding: 10px;"> For on-demand scanning of files and directories, it is recommended that you use the <code>scan</code> command.</div> <p>Arguments</p> <p><code><path></code>—path to the file or directory which is selected to be scanned.</p> <p>Options</p> <p><code>--ScanTimeout <number></code>—specify time-out to scan one file, in ms.</p> <p>If the value is set to 0, time on scanning is not limited. Default value: 0.</p>



Command	Description
	<p><code>--HeuristicAnalysis <On Off></code>—enable or disable heuristic analysis during the scanning.</p> <p>Default value: On.</p> <p><code>--PackerMaxLevel <number></code>—set the maximum nesting level when scanning packed objects. A packed object is executable code compressed with special software (UPX, PELock, PECompact, Petite, ASPack, Morphine and so on). Such objects may include other packed objects which may also include packed objects. etc. The value of this parameter specifies the nesting limit beyond which packed objects inside other packed objects will not be scanned.</p> <p>If the value is set to 0, nested objects will be skipped during scanning.</p> <p>Default value: 8.</p> <p><code>--ArchiveMaxLevel <number></code>—set the maximum nesting level when scanning archives (zip, rar, and so on) in which other archives may be enclosed (and these archives may also include other archives, and so on). The value of this parameter specifies the nesting limit beyond which archives enclosed in other archives will not be scanned.</p> <p>If the value is set to 0, nested objects will be skipped during scanning.</p> <p>Default value: 8.</p> <p><code>--MailMaxLevel <number></code>—set the maximum nesting level when scanning files of mailers (pst, tbb and so on) in which other files may be enclosed (and these files may also include other files and so on). The value of this parameter specifies the nesting limit beyond which objects inside other objects will not be scanned.</p> <p>If the value is set to 0, nested objects will be skipped during scanning.</p> <p>Default value: 8.</p> <p><code>--ContainerMaxLevel <number></code>—set the maximum nesting level when scanning other types objects inside which other objects are enclosed (HTML pages, jar-files, etc.). The value of this parameter specifies the nesting limit beyond which objects inside other objects will not be scanned.</p> <p>If the value is set to 0, nested objects will be skipped during scanning.</p> <p>Default value: 8.</p> <p><code>--MaxCompressionRatio <ratio></code>—set the maximum compression ratio of scanned objects.</p> <p>The ratio must be at least equal to 2.</p> <p>Default value: 3000.</p> <p><code>--OnKnownVirus <action></code>—an action to perform upon detection of a known threat by using signature-based analysis.</p> <p>Possible actions: Report, Cure, Quarantine, Delete.</p> <p>Default value: Report.</p> <p><code>--OnIncurable <action></code>—an action to perform upon detection an incurable threat or when curing action (Cure) failed.</p> <p>Possible actions: Report, Quarantine, Delete.</p>



Command	Description
	<p>Default value: Report.</p> <p>--OnSuspicious <i><action></i>—an action to perform upon detection of a suspicious object by heuristic analysis.</p> <p>Possible actions: Report, Quarantine, Delete.</p> <p>Default value: Report.</p> <p>--OnAdware <i><action></i>—an action to perform upon detection of adware programs.</p> <p>Possible actions: Report, Quarantine, Delete.</p> <p>Default value: Report.</p> <p>--OnDialers <i><action></i>—an action to perform upon detection of a dialer.</p> <p>Possible actions: Report, Quarantine, Delete.</p> <p>Default value: Report.</p> <p>--OnJokes <i><action></i>—an action to perform upon detection of joke software.</p> <p>Possible actions: Report, Quarantine, Delete.</p> <p>Default value: Report.</p> <p>--OnRiskware <i><action></i>—an action to perform upon detection of riskware.</p> <p>Possible actions: Report, Quarantine, Delete.</p> <p>Default value: Report.</p> <p>--OnHacktools <i><action></i>—an action to perform upon detection of a hacktool.</p> <p>Possible actions: Report, Quarantine, Delete.</p> <p>Default value: Report.</p> <div data-bbox="608 1370 1449 1563" style="background-color: #e6f2e6; padding: 10px;"> If threat is detected in a file placed in a container (an archive, an email message, and so on), instead of removing the file (Delete), the tool moves the container to the quarantine (Quarantine).</div>
rawscan <i><path></i>	<p>Purpose: start “raw” scanning of the specified file or directory by Dr.Web Scanning Engine directly, without the use of Dr.Web File Checker.</p>



Command	Description
	<div data-bbox="635 293 703 353"></div> <p data-bbox="746 293 1417 389">Note that threats detected by “raw” scanning are not included into the list of detected threats that is displayed by the <code>threats</code> command (see below).</p> <hr/> <p data-bbox="746 456 1433 719">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 <i>one</i> threat is neutralized of those threats that are detected in a file (not <i>all</i> threats might be neutralized). Thus, it is <i>not recommended</i> to use this command if you need thorough file scanning. In the latter case it is recommended to use the <code>scan</code> command.</p> <p data-bbox="571 786 703 815">Arguments</p> <p data-bbox="571 842 1385 871"><path>—path to the file or directory which is selected to be scanned.</p> <p data-bbox="571 898 667 927">Options</p> <p data-bbox="571 954 1449 1050"><code>--ScanEngine <path></code>—path to the UNIX socket of the Dr.Web Scanning Engine. If not specified, an autonomous instance of the scan engine is started (which will be shut down once the scanning is completed).</p> <p data-bbox="571 1077 1209 1106"><code>--Report <type></code>—specifies the type of scan report.</p> <p data-bbox="608 1122 794 1151">Allowed values:</p> <ul data-bbox="608 1167 1134 1285" style="list-style-type: none">• BRIEF—brief report.• DEBUG—detailed report.• JSON—a serialized report in JSON format. <p data-bbox="608 1308 858 1337">Default value: BRIEF.</p> <p data-bbox="571 1352 1401 1382"><code>--ScanTimeout <number></code>—specify time-out to scan one file, in ms.</p> <p data-bbox="608 1397 1241 1426">If the value is set to 0, time on scanning is not limited.</p> <p data-bbox="608 1442 794 1471">Default value: 0.</p> <p data-bbox="571 1487 1449 1727"><code>--PackerMaxLevel <number></code>—set the maximum nesting level when scanning packed objects. A packed object is executable code compressed with special software (UPX, PELock, PECompact, Petite, ASPack, Morphine and so on). Such objects may include other packed objects which may also include packed objects. etc. The value of this parameter specifies the nesting limit beyond which packed objects inside other packed objects will not be scanned.</p> <p data-bbox="608 1749 1433 1778">If the value is set to 0, nested objects will be skipped during scanning.</p> <p data-bbox="608 1794 794 1823">Default value: 8.</p> <p data-bbox="571 1839 1449 2002"><code>--ArchiveMaxLevel <number></code>—set the maximum nesting level when scanning archives (zip, rar, and so on) in which other archives may be enclosed (and these archives may also include other archives, and so on). The value of this parameter specifies the nesting limit beyond which archives enclosed in other archives will not be scanned.</p>



Command	Description
	<p>If the value is set to 0, nested objects will be skipped during scanning. Default value: 8.</p> <p><code>--MailMaxLevel <number></code>—set the maximum nesting level when scanning files of mailers (pst, tbb and so on) in which other files may be enclosed (and these files may also include other files and so on). The value of this parameter specifies the nesting limit beyond which objects inside other objects will not be scanned.</p> <p>If the value is set to 0, nested objects will be skipped during scanning. Default value: 8.</p> <p><code>--ContainerMaxLevel <number></code>—set the maximum nesting level when scanning other types objects inside which other objects are enclosed (HTML pages, jar-files, etc.). The value of this parameter specifies the nesting limit beyond which objects inside other objects will not be scanned.</p> <p>If the value is set to 0, nested objects will be skipped during scanning. Default value: 8.</p> <p><code>--MaxCompressionRatio <ratio></code>—set the maximum compression ratio of scanned objects.</p> <p>The ratio must be at least equal to 2. Default value: 3000.</p> <p><code>--MaxSizeToExtract <size></code>—<i>specify the maximum size for files enclosed in archives. Files whose size is greater than the value of this parameter will be skipped when scanning. There is no size limit for files in archives by default. The size is specified as a number with a suffix (b, kb, mb, gb). If no suffix is specified, the value is treated as size in bytes.</i></p> <p>Default value: none.</p> <p><code>--HeuristicAnalysis <On Off></code>—enable or disable heuristic analysis during the scanning. Default value: On.</p> <p><code>--Cure <Yes No></code>—enable or disable attempts to cure detected threats.</p> <p>If the value is set to No, only a notification about a detected threat is displayed. Default value: No.</p> <p><code>--ListCleanItem</code>—enable outputting the list of clean (non-infected) files found inside a container that was scanned.</p> <p><code>--ShellTrace</code>—enable display of additional debug information when scanning a file.</p> <p><code>--Output <path to file></code>—duplicate the output of the command to the specified file</p>
<code>remotescan <host> <path></code>	<p>Purpose: initiates scanning of the specified file or directory on the specified remote host by connecting to it via <i>SSH</i> or <i>Telnet</i>.</p>



Command	Description
	<div data-bbox="630 286 699 353" style="float: left; margin-right: 10px;"></div> <div data-bbox="746 286 1433 425"><p>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 <code>threats</code> command (see below).</p><hr/></div> <div data-bbox="746 488 1433 855"><p>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, set-top boxes, 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, and so on), or via running an anti-virus software installed on them.</p></div> <div data-bbox="571 918 705 949">Arguments</div> <div data-bbox="571 976 1417 1079"><ul style="list-style-type: none">• <code><host></code>—IP address or a domain name of the remote host.• <code><path></code>—path to the file or directory to be scanned (the path must be absolute).</div> <div data-bbox="571 1111 667 1142">Options</div> <div data-bbox="571 1169 1343 1232"><p><code>-m [--Method] <SSH Telnet></code>—remote host connection method (protocol).</p></div> <div data-bbox="603 1245 1062 1276"><p>If method is not specified, SSH is used.</p></div> <div data-bbox="571 1290 1417 1352"><p><code>-l [--Login] <name></code>—login (user name) used for authorization on the remote host via the selected protocol.</p></div> <div data-bbox="603 1366 1439 1433"><p>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.</p></div> <div data-bbox="571 1447 1439 1518"><p><code>-i [--Identity] <path to file></code>—path to the file containing a private key used for authentication of the specified user via the selected protocol.</p></div> <div data-bbox="571 1532 1417 1599"><p><code>-p [--Port] <number></code>—number of the port on the remote host for connecting via the selected protocol.</p></div> <div data-bbox="603 1612 1449 1680"><p>Default value: default port for the selected protocol (22 for SSH, 23 for Telnet).</p></div> <div data-bbox="571 1693 1407 1760"><p><code>--ForceInteractive</code>—use the SSH interactive session (only for SSH connections).</p></div> <div data-bbox="603 1774 798 1805"><p><i>Optional feature.</i></p></div> <div data-bbox="571 1818 1369 1890"><p><code>--TransferListenAddress <address></code>—an address, listened to receive files transferred from the remote device for scanning.</p></div> <div data-bbox="603 1904 1340 1935"><p>Optional feature. If not indicated, an arbitrary address is used.</p></div> <div data-bbox="571 1948 1369 2016"><p><code>--TransferListenPort <port></code>—a port, listened to receive files transferred from the remote device for scanning.</p></div>



Command	Description
	<p>Optional feature. If not indicated, an arbitrary port is used.</p> <p><code>--TransferExternalAddress <address></code>—an address specified to the remote device to send files for scanning.</p> <p>Optional feature. If not indicated, use the <code>--TransferListenAddress</code> value, or the outgoing address of the already established session.</p> <p><code>--TransferExternalPort <port></code>—a port to transfer files for scanning, specified for the remote device.</p> <p>Optional feature. If not indicated, an automatically determined port is used.</p> <p><code>--Password <password></code>—password used for authentication of a user via the selected protocol.</p> <p>Please note that the password is transferred as a plain text.</p> <p><code>--Exclude <path></code>—the path to be excluded from scanning. The path can contain a file mask with the following allowed symbols: '?' and '*', as well as the symbol classes '[]', '[!]', '[^]'. The path (including the path with the file mask) must be absolute.</p> <p>Facultative option; can be set more than once.</p> <p><code>--Report <type></code>—specifies the type of scan report.</p> <p>Allowed values:</p> <ul style="list-style-type: none">• BRIEF—brief report.• DEBUG—detailed report.• JSON—a serialized report in JSON format. <p>Default value: BRIEF.</p> <p><code>--ScanTimeout <number></code>—specify time-out to scan one file, in ms.</p> <p>If the value is set to 0, time on scanning is not limited.</p> <p>Default value: 0.</p> <p><code>--PackerMaxLevel <number></code>—set the maximum nesting level when scanning packed objects. A packed object is executable code compressed with special software (UPX, PELock, PECompact, Petite, ASPack, Morphine and so on). Such objects may include other packed objects which may also include packed objects. etc. The value of this parameter specifies the nesting limit beyond which packed objects inside other packed objects will not be scanned.</p> <p>If the value is set to 0, nested objects will be skipped during scanning.</p> <p>Default value: 8.</p> <p><code>--ArchiveMaxLevel <number></code>—set the maximum nesting level when scanning archives (zip, rar, and so on) in which other archives may be enclosed (and these archives may also include other archives, and so on). The value of this parameter specifies the nesting limit beyond which archives enclosed in other archives will not be scanned.</p> <p>If the value is set to 0, nested objects will be skipped during scanning.</p>



Command	Description
	<p>Default value: 8.</p> <p><code>--MailMaxLevel <number></code>—set the maximum nesting level when scanning files of mailers (pst, tbb and so on) in which other files may be enclosed (and these files may also include other files and so on). The value of this parameter specifies the nesting limit beyond which objects inside other objects will not be scanned.</p> <p>If the value is set to 0, nested objects will be skipped during scanning.</p> <p>Default value: 8.</p> <p><code>--ContainerMaxLevel <number></code>—set the maximum nesting level when scanning other types objects inside which other objects are enclosed (HTML pages, jar-files, etc.). The value of this parameter specifies the nesting limit beyond which objects inside other objects will not be scanned.</p> <p>If the value is set to 0, nested objects will be skipped during scanning.</p> <p>Default value: 8.</p> <p><code>--MaxCompressionRatio <ratio></code>—set the maximum compression ratio of scanned objects.</p> <p>The ratio must be at least equal to 2.</p> <p>Default value: 3000.</p> <p><code>--MaxSizeToExtract <size></code>—<i>specify the maximum size for files enclosed in archives. Files whose size is greater than the value of this parameter will be skipped when scanning. There is no size limit for files in archives by default. The size is specified as a number with a suffix (b, kb, mb, gb). If no suffix is specified, the value is treated as size in bytes.</i></p> <p>Default value: none.</p> <p><code>--HeuristicAnalysis <On Off></code>—enable or disable heuristic analysis during the scanning.</p> <p>Default value: On</p>

3.2. Commands to manage updates and operation in the centralized protection mode

The following commands for managing updates are available, as well as commands for operation in the centralized protection mode:

Command	Description
update	<p>Purpose: initiates updates of anti-virus components (virus databases, the scan engine, etc., depending on the distribution) from the Doctor Web update servers or the local cloud via Dr.Web MeshD, terminates the updating process if already running, or performs rollback of the latest update to previous versions of the updated files.</p>



Command	Description
	<div data-bbox="611 259 1449 412" style="background-color: #e6f2e6; padding: 10px;"> The command has no effect if Dr.Web for UNIX Internet Gateways is connected to the centralized protection server.</div> <p>Arguments: none.</p> <p>Options</p> <p><code>-l [--local-cloud]</code> uses the local cloud connected to Dr.Web for UNIX Internet Gateways to download the updates. If the option is not specified, the updates are downloaded from the Doctor Web update servers (default behavior).</p> <p><code>--From <path></code>—apply updates from a specified directory offline.</p> <p><code>--Path <path></code>—store files for updating offline in a specified directory; if this directory already has files, then they will be updated.</p> <p><code>--Rollback</code>—rollback the last update, and restore the previous version of files that have been updated during the last update.</p> <p><code>--Stop</code>—terminate the running updating process</p>
<code>esconnect</code> <code><server> [: <port>]</code>	<p>Purpose: connects Dr.Web for UNIX Internet Gateways to the specified centralized protection server (for example, Dr.Web Enterprise Server). For details on the operation modes, refer to the Operation Modes.</p> <p>Arguments</p> <ul style="list-style-type: none">• <code><server></code>—IP address or network name of the host on which the centralized protection server is operating. This argument is mandatory.• <code><port></code>—port number used by the centralized protection server. The argument is optional and should be specified only if the centralized protection server uses a non-standard port. <p>Options</p> <p><code>--Certificate <path></code>—a file path to a certificate of the centralized protection server, the connection to which will be established.</p> <p><code>--Login <ID></code>—login (workstation identifier) used for connection to the centralized protection server.</p> <p><code>--Password <password></code>—password for connection to the centralized protection server.</p> <p><code>--Group <ID></code>—identifier of the group to which the workstation is added on connection.</p> <p><code>--Rate <ID></code>—identifier of the tariff group applied to your workstation when it is included in one of the centralized protection server groups (can be specified only together with the <code>--Group</code> option).</p> <p><code>--Compress <On Off></code>—enables (On) or disables (Off) forced compression of transmitted data. If not specified, usage of compression is determined by the server.</p>



Command	Description
	<p><code>--Encrypt <On Off></code>—enables (On) or disables (Off) forced encryption of transmitted data. If not specified, usage of encryption is determined by the server.</p> <p><code>--Newbie</code>—connect as a “newbie” (get a new account on the server).</p> <div style="background-color: #e6f2e6; padding: 10px;"> This command requires <code>drweb-ctl</code> to be started with <i>root</i> privileges. If necessary, use the <code>su</code> or <code>sudo</code> commands.</div>
<code>esdisconnect</code>	<p>Purpose: disconnect Dr.Web for UNIX Internet Gateways from the centralized protection server and switch its operation to standalone mode.</p> <div style="background-color: #e6f2e6; padding: 10px;"> The command has no effect if Dr.Web for UNIX Internet Gateways already operates in standalone mode.</div> <p>Arguments: none.</p> <p>Options: none.</p> <div style="background-color: #e6f2e6; padding: 10px;"> This command requires <code>drweb-ctl</code> to be started with <i>root</i> privileges. If necessary, use the <code>su</code> or <code>sudo</code> commands.</div>

3.3. Configuration Management Commands

The following commands to manage configuration are available:

Command	Description
<code>cfset</code> <code><section> . <parameter></code> <code><value></code>	<p>Purpose: to change the active value of the specified parameter in the current configuration of Dr.Web for UNIX Internet Gateways.</p> <p>Arguments</p> <ul style="list-style-type: none">• <code><section></code>—name of the configuration file section where the parameter resides. This argument is mandatory.• <code><parameter></code>—name of the parameter. The argument is mandatory.• <code><value></code>—new parameter value. This argument is mandatory.



Command	Description
	<div data-bbox="619 259 1449 922" style="background-color: #e6f2e6; padding: 10px;"> To specify the parameter value, we use the format <code><section>.<parameter> <value></code>. Assignment character '=' is not used here. Note that if you want to indicate several parameter values, you need to repeatedly call the <code>cfset</code> command, as many times as the number of parameter values you want to add. To add a new value to the list of parameter values, you need to use the <code>-a</code> option (see below). You cannot specify the string <code><parameter> <value 1>, <value 2></code> as an argument, because the string "<code><value 1>, <value 2></code>" will be considered one value of the <code><parameter></code>. For description of the configuration file, refer to the section Appendix D. Dr.Web for UNIX Internet Gateways Configuration File, as well as the documentation page displayed by <code>man 5 drweb.ini</code>.</div> <p data-bbox="571 958 676 992">Options</p> <p data-bbox="571 1014 1449 1149"><code>-a [--Add]</code>—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.</p> <p data-bbox="571 1167 1449 1267"><code>-e [--Erase]</code>—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).</p> <p data-bbox="571 1285 1369 1352"><code>-r [--Reset]</code>—reset the parameter value to the default. At that, <code><value></code> is not required in the command and is ignored if specified.</p> <p data-bbox="571 1370 1436 1471">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.</p> <p data-bbox="571 1505 1449 1639">For sections that describe the individual parameters for connection points of Dr.Web ClamD, using the <code>-r</code> option changes the parameter value in the individual settings section to the value of its parent parameter in the settings of this component.</p> <p data-bbox="571 1673 1453 1740">If you need to add a new connection point <code><point></code> for Dr.Web ClamD, use the following command:</p> <pre data-bbox="616 1767 1259 1827">cfset ClamD.Endpoint.<point> -a, for example: cfset ClamD.Endpoint.point1 -a</pre> <div data-bbox="619 1854 1449 2007" style="background-color: #e6f2e6; padding: 10px;"> This command requires <code>drweb-ctl</code> to be started with root privileges. If necessary, use the <code>su</code> or <code>sudo</code> commands.</div>



Command	Description
<pre>cfshow [<section> [. <parameter>]]</pre>	<p>Purpose: displays parameters of the current configuration of Dr.Web for UNIX Internet Gateways.</p> <p>The command to display parameters is specified as follows <section>.<parameter> = <value>. Sections and parameters of non-installed components are not displayed.</p> <p>Arguments</p> <ul style="list-style-type: none">• <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. Optional argument. 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. <p>Options</p> <p>--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.</p> <p>--Changed—display only those parameters whose values differ from the default ones.</p> <p>--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).</p> <p>--Value—display only value of the specified parameter (the <parameter> argument is mandatory in this case)</p>
<pre>reload</pre>	<p>Purpose: send the SIGHUP signal to the Dr.Web ConfigD configuration daemon.</p> <p>On receiving this signal, the Dr.Web ConfigD configuration daemon rereads the configuration and sends its changes to the Dr.Web for UNIX Internet Gateways components. Then, the configuration daemon reopens the Dr.Web for UNIX Internet Gateways log, restarts the components that use virus databases (including the scan engine), and attempts to restart those components which were terminated abnormally.</p> <p>Arguments: none.</p> <p>Options: none</p>



3.4. Commands to Manage Detected Threats and Quarantine

The following commands for managing threats and quarantine are available:

Command	Description
<code>threats</code> [<action> <object>]	<p>Purpose: apply the specified action to detected threats, selected by their identifiers. Type of the action is specified by the command option.</p> <p>If the action is not specified, displays information on detected but not neutralized threats. The information on threats is displayed according the format, specified using the optional <code>--Format</code> option. If the <code>--Format</code> option is not specified, for each threat the following information is displayed:</p> <ul style="list-style-type: none">• 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, and so on. <p>Arguments: none.</p> <p>Options</p> <p><code>--Format</code> "<format string>"—displays information on threats in the specified format. The description of format string is below.</p> <p>If this option is specified along with any action options, it is ignored.</p> <p><code>-f</code> [<code>--Follow</code>]—wait for new messages about new threats and display them once they are received (CTRL+C interrupts the waiting).</p> <p>If this option is specified along with any action options, it is ignored.</p> <p><code>--Directory</code> <list of directories>—displays only threats detected in files in directories from <list of directories>.</p> <p>If this option is applied along with any options mentioned below, it is ignored.</p> <p><code>--Cure</code> <threat list>—attempt to cure the listed threats (list threat identifiers separating them with commas).</p> <p><code>--Quarantine</code> <threat list> moves the listed threats to quarantine (list threat identifiers are separated with commas).</p> <p><code>--Delete</code> <threat list>—delete the listed threats (list threat identifiers separating them with commas).</p> <p><code>--Ignore</code> <threat list>—ignore the listed threats (list threat identifiers separating them with commas).</p>



Command	Description
	<p>If you need to apply the action to all detected threats, specify <code>All</code> instead of <code><threat list></code>. For example, the command:</p> <pre data-bbox="576 353 1449 427">\$ drweb-ctl threats --Quarantine All</pre> <p>moves all detected malicious objects to quarantine</p>
<code>quarantine</code> [<code><action></code> <code><object></code>]	<p>Purpose: applies an action to the specified object in quarantine.</p> <p>If an action is not specified, information on quarantined objects and their identifiers together with brief information on original files moved to quarantine is displayed. Information on isolated objects is displayed according a format, specified with optional <code>--Format</code> argument. If the <code>--Format</code> argument is not specified, for every isolated (quarantined) object the following information is displayed:</p> <ul style="list-style-type: none">• 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). <p>Arguments: none.</p> <p>Options</p> <p><code>-a</code> [<code>--Autonomous</code>] starts a separate instance of the Dr.Web File Checker file scanning component to perform the specified quarantine command and terminate it upon completion.</p> <p>This option can be applied along with any options mentioned below.</p> <p><code>--Format</code> "<code><format string></code>"—displays information on quarantined objects in the specified format. The description of format string is below.</p> <p>If this option is specified along with any action options, it is ignored.</p> <p><code>-f</code> [<code>--Follow</code>]—wait for new messages about new threats and display them once they are received (CTRL+C interrupts the waiting).</p> <p>If this option is specified along with any action options, it is ignored.</p> <p><code>--Discovery</code> [<code><list of directories></code>,] searches for quarantine directories in the specified list of directories and add them to the consolidated quarantine upon detecting a threat. If the <code><list of directories></code> is not specified, it searches for quarantine directories in the common locations of the file system (volume mounting points and user home directories).</p>



Command	Description
	<p>This option can be specified not only with the <code>-a (--Autonomous)</code> option (see above), but also with any options/actions listed below. Moreover, if the <code>quarantine</code> command is launched as an autonomous copy, that is, with the <code>-a (--Autonomous)</code> option but without the <code>--Discovery</code> option, then it is equivalent to the call of:</p> <pre data-bbox="612 443 1449 510">quarantine --Autonomous --Discovery</pre> <p><code>--Delete <object></code>—delete the specified object from quarantine.</p> <p>Note that objects are deleted from quarantine permanently—this action is irreversible.</p> <p><code>--Cure <object></code>—try to cure the specified object in the quarantine.</p> <p>Note that even if the object was successfully cured, it will remain in quarantine. To restore the cured object from quarantine, use the <code>--Restore</code> option.</p> <p><code>--Restore <object></code>—restore the specified object from the quarantine to its original location.</p> <p>Note that this command may require <code>drweb-ctl</code> to be started with root privileges. You can restore the file from quarantine even if it is infected.</p> <p><code>--TargetPath <path></code>—restores an object from the quarantine to the specified location: either as a file with the name specified here (if the <code><path></code> is a path to a file), or just to the specified directory (if the <code><path></code> is a path to a directory). A path can be an absolute as well as relative (referring to a current directory).</p> <p>Note that this option can only be used in combination with the <code>--Restore</code> option.</p> <p>As an <code><object></code>, specify the object identifier in quarantine. To apply the action to all quarantined objects, specify <code>All</code> instead of <code><object></code>. For example, the command:</p> <pre data-bbox="571 1413 1449 1514">\$ drweb-ctl quarantine --Restore All --TargetPath test</pre> <p>restores all quarantined objects and puts them in <code>test</code> subdirectory, located in a current directory, from which <code>drweb-ctl</code> command was launched.</p> <p>Note that for the <code>--Restore All</code> variant the additional option <code>--TargetPath</code>, if specified, must set a path to a directory, not a path to a file</p>

Formatted output for threats and quarantine commands



The output format is defined using the format string, specified as the optional argument `--Format`. The format string must be specified in quotes. The format string can include common symbols (displayed "as is"), as well as special markers, output as certain information. The following markers are available:

1. Common for `threats` and `quarantine` commands:

Marker	Description
<code>%{n}</code>	New string
<code>%{t}</code>	Tabulation
<code>%{threat_name}</code>	The name of detected threat (virus) according Doctor Web classification
<code>%{threat_type}</code>	Threat Type ("known virus", and so on) according Doctor Web classification
<code>%{size}</code>	Original file size
<code>%{origin}</code>	The full name of the original file with path
<code>%{path}</code>	Synonym for <code>%{origin}</code>
<code>%{ctime}</code>	Date/time of original file modifying in "%Y-%b-%d %H:%M:%S" format (for example, "2018-Jul-20 15:58:01")
<code>%{timestamp}</code>	Similar to <code>%{ctime}</code> , but in the <i>UNIX timestamp</i> format
<code>%{owner}</code>	Username of the original file owner
<code>%{rowner}</code>	The remote user owner of the original file (if not applicable or value is unknown it is replaced with ?)

2. Specific for `threats` command:

Marker	Description
<code>%{hid}</code>	The identifier of the threat record in the history of events associated with the threat
<code>%{tid}</code>	The threat identifier
<code>%{htime}</code>	Date/time of the event related to a threat
<code>%{app}</code>	The identifier of the Dr.Web for UNIX Internet Gateways component which processed a threat
<code>%{event}</code>	The latest event related to a threat: <ul style="list-style-type: none">• <code>FOUND</code>—a threat was detected;• <code>Cure</code>—a threat was cured;• <code>Quarantine</code>—a file with threat was moved to quarantine;• <code>Delete</code>—a file with threat was deleted;



Marker	Description
	<ul style="list-style-type: none">Ignore—a threat was ignored;RECAPTURED—a threat was detected again by an other component
<code>%{err}</code>	Error message text (if no error is replaced with an empty string)

3. Specific for quarantine command:

Marker	Description
<code>%{qid}</code>	The identifier of quarantined object
<code>%{qtime}</code>	Date/time of moving the object to quarantine
<code>%{curetime}</code>	Date/time of curing attempt of the object moved to quarantine (if not applicable or value is unknown it is replaced with ?)
<code>%{cures}</code>	The result of the quarantined object curing attempt: <ul style="list-style-type: none">cured—a threat is cured;not cured—a threat was not cured or no curing attempts were performed

Example

```
$ drweb-ctl quarantine --Format "{%{n} %{origin}: %{threat_name} - %{qtime}%{n}}"
```

This command displays quarantine contents as records of the following type:

```
{  
  <path to file>: <threat name> - <date of moving to quarantine>  
}  
...
```

3.5. Information Commands

The following information commands are available:

Command	Description
<code>appinfo</code>	<p>Purpose: output information on the active Dr.Web for UNIX Internet Gateways components.</p> <p>The following information is displayed about each component that is currently running:</p> <ul style="list-style-type: none">Internally-used name.Process identifier GNU/Linux (PID).State (running, stopped, and so on).



Command	Description
	<ul style="list-style-type: none">• Error code, if the work of the component has been terminated because of an error.• Additional information (optional). <p>For the configuration daemon (<code>drweb-configd</code>) the following is displayed as additional information:</p> <ul style="list-style-type: none">• The list of installed components—<i>Installed</i>.• The list of components which must be launched by the configuration daemon—<i>Should run</i>. <p>Arguments: none.</p> <p>Options</p> <p><code>-f [--Follow]</code>—waits for new messages on module status change and display them once such a message is received (CTRL+C interrupts the waiting)</p>
<code>baseinfo</code>	<p>Purpose: display the information on the current version of the scan engine and status of virus databases.</p> <p>The following information is displayed:</p> <ul style="list-style-type: none">• Version of the scan 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 scan engine.• The time of the next scheduled automatic update. <p>Arguments: none.</p> <p>Options</p> <p><code>-l [--List]</code>—displays the full list of downloaded files of virus databases and number of virus records in each file</p>
<code>certificate</code>	<p>Purpose: displays contents of the trusted Dr.Web certificate used by Dr.Web for UNIX Internet Gateways. To save the certificate to the <code><cert_name>.pem</code> file, you can use the following command:</p> <pre>\$ drweb-ctl certificate > <cert_name>.pem</pre> <p>Arguments: none.</p> <p>Options: none</p>
<code>events</code>	<p>Purpose: viewing of the Dr.Web for UNIX Internet Gateways events. Apart from that, the command allows you to manage events (mark as read, remove).</p>



Command	Description
	<p>Arguments: none.</p> <p>Options</p> <p><code>--Report <type></code>—specify the type of event report.</p> <p>Allowed values:</p> <ul style="list-style-type: none">• BRIEF—brief report.• DEBUG—detailed report.• JSON—a serialized report in JSON format. <p><code>-f [--Follow]</code>—waits for new events and displays them upon emergence (CTRL+C interrupts the standby).</p> <p><code>-s [--Since] <date, time></code>—shows the events that occurred before the specified timestamp (<date, time> is specified as YYYY-MM-DD hh:mm:ss).</p> <p><code>-u [--Until] <date, time></code>—shows the events that occurred no later than the specified timestamp (<date, time> is specified as YYYY-MM-DD hh:mm:ss).</p> <p><code>-t [--Types] <type list></code>—shows only events of the specified types (separated by commas).</p> <p>The following event types are available:</p> <ul style="list-style-type: none">• Mail—indicates that a threat has been detected in an email;• UnexpectedAppTermination—unexpected shutdown of a component. <p>To view all types of events, use <code>All</code>.</p> <p><code>--ShowSeen</code>—displays of already read events as well.</p> <p><code>--Show <list of events></code>—displays the listed events (event identifiers are separated by commas).</p> <p><code>--Delete <list of events></code>—removal of listed events (event identifiers are separated by commas).</p> <p><code>--MarkAsSeen <list of events></code>—marks the listed events as read (event identifiers are separated with a comma).</p> <p>If you want to mark as “read” or delete all events, specify <code>All</code> instead of <events list>. For example, the command:</p> <pre data-bbox="568 1639 1449 1711">\$ drweb-ctl events --MarkAsSeen All</pre> <p>will mark as “read” all existing events</p>
<code>report <type></code>	<p>Purpose: create a report on Dr.Web for UNIX Internet Gateways events in the HTML format (the page body is output to the specified file).</p>



Command	Description
	<p>Arguments</p> <p><i><type></i>—event type that required reporting (indicate one type). See possible values in the <code>--Types</code> option description of the <code>events</code> command above. A mandatory argument.</p> <p>Options</p> <p><code>-o [--Output] <path to file></code>—save the report to the specified file. The option is mandatory.</p> <p><code>-s [--Since] <date, time></code>—reports events that occurred no earlier than the specified timestamp (<i><date, time></i> is specified as <code>YYYY-MM-DD hh:mm:ss</code>).</p> <p><code>-u [--Until] <date, time></code>—reports the events that occurred no later than the specified timestamp (<i><date, time></i> is specified as <code>YYYY-MM-DD hh:mm:ss</code>).</p> <p><code>--TemplateDir <path to directory></code>—a path to the directory that contains HTML report templates.</p> <p>Options <code>-s</code>, <code>-u</code>, and <code>--TemplateDir</code> are not mandatory. For example, the following command:</p> <pre>\$ drweb-ctl report Mail -o report.html</pre> <p>generates a report on all existing email message threat detection events, based on the default template, and saves the result in the <code>report.html</code> file in the current directory</p>
license	<p>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 website).</p> <p>If no options are specified, then the following information is displayed (if you are using a license for the standalone mode):</p> <ul style="list-style-type: none">• License number.• Date and time when the license expires. <p>If you are using a license provided to you by the centralized protection server (for the use of the product in the centralized protection mode or in the mobile mode), then the appropriate message will be displayed.</p> <p>Arguments: none.</p> <p>Options</p> <p><code>--GetRegistered <serial number></code>—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 centralized protection mode, when the license is managed by a centralized protection server).</p>



Command	Description
	<p>If the serial number is not the one provided for the demo period, you must first register it at the company website.</p> <pre>--Proxy http://<username>:<password>@<server address>:<port number>—get a license key file through a proxy server (used only with the --GetRegistered option).</pre> <p>For further information on licensing of Dr.Web products, refer to the Licensing section.</p> <div data-bbox="608 562 1449 680" style="background-color: #e6f2e6; padding: 10px;"> To register a serial number, an internet connection is required.</div>
log	<p>Purpose: displays the latest log records of Dr.Web for UNIX Internet Gateways on console screen (in the <i>stdout</i> thread), similar to <code>tail</code> command.</p> <p>Arguments: none.</p> <p>Options</p> <pre>-s [--Size] <number>—the number of the last log records that are to be displayed on a screen.</pre> <pre>-c [--Components] <components list>—the list of component identifiers, which records are displayed. Identifiers are defined with comma separation. If the argument is not defined, all available records logged by all components are displayed.</pre> <p>Actual identifiers of the components installed (e.g. internal components names, displayed in log) you can define by using the <code>appinfo</code> command (see above).</p> <pre>-f [--Follow]—waits for new messages in log and display them once such a message is received (interrupt waiting by pressing CTRL+C)</pre>
stat	<p>Purpose: Output statistics about the operation of components that process files or about the operation of the network data scanning agent Dr.Web Network Checker (pressing CTRL+C or Q interrupts the statistics display).</p> <p>The statistics output includes:</p> <ul style="list-style-type: none">• 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.• usage percentage of the scanned files cache.• average number of scan errors per second. <p>For the distributed scanning agent, the following information is output:</p> <ul style="list-style-type: none">• list of local clients that initiated scanning.• list of remote hosts that received files for scanning.



Command	Description
	<ul style="list-style-type: none">• list of remote hosts that sent files for scanning. <p>For local clients of the distributed scanning agent, their PID and name are specified; for remote clients—address and port of the host.</p> <p>For both clients—local and remote—the following information is output:</p> <ul style="list-style-type: none">• average number of files scanned per second.• average number of sent and received bytes per second.• average number of errors per second. <p>Arguments: none.</p> <p>Options</p> <p><code>-n [--netcheck]</code> outputs statistics on operation of the network data scanning agent</p>

Usage Examples

This section contains examples of using the Dr.Web Ctl (`drweb-ctl`) utility:

- Object Scanning:
 - [Simple Scanning Commands](#)
 - [Scanning of Files Selected by Criteria](#)
 - [Scanning of Additional Objects](#)
- [Configuration Management](#)
- [Threats Management](#)
- [An Example of Operation in the Autonomous Copy Mode](#)

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` drive:

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

4. Perform scanning of the running processes:



```
$ drweb-ctl procsan
```

1.2. Scanning of Files Selected by Criteria

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

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 host `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 Dr.Web for UNIX Internet Gateways package, including information about running components:

```
$ 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 `[LinuxSpider]` section of the active configuration (this will disable the SpIDer Guard file system monitor):

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

4. Force update of anti-virus components of Dr.Web for UNIX Internet Gateways:

```
$ drweb-ctl update
```

5. Restart the component configuration of Dr.Web for UNIX Internet Gateways:

```
# 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
```



6. Connect Dr.Web for UNIX Internet Gateways to the [centralized protection](#) server operating on host `192.168.0.1` if the server certificate is located in the file `/home/user/cscert.pem`:

```
$ drweb-ctl esconnect 192.168.0.1 --Certificate /home/user/cscert.pem
```

7. Connect Dr.Web for UNIX Internet Gateways to the [centralized protection](#) server using the `settings.cfg` configuration file:

```
$ drweb-ctl esconnect --cfg <path to the settings.cfg file>
```

8. Disconnecting Dr.Web for UNIX Internet Gateways from the centralized protection server:

```
# drweb-ctl esdisconnect
```

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 esdisconnect
```

9. View the last log records made by the `drweb-update` and `drweb-configd` components in the Dr.Web for UNIX Internet Gateways log:

```
# drweb-ctl log -c Update,ConfigD
```

3. Threats Management

1. Display information on detected threats:

```
$ drweb-ctl threats
```

2. Move all files containing threats which were not neutralized to quarantine:

```
$ drweb-ctl threats --Quarantine All
```

3. Display list of files moved to quarantine:

```
$ drweb-ctl quarantine
```

4. Restore all files from quarantine:

```
$ drweb-ctl quarantine --Restore All
```

4. An Example of Operation in the Autonomous Copy Mode

1. Scan files and process quarantine in the autonomous copy mode:

```
$ drweb-ctl scan /home/user -a --OnKnownVirus=Quarantine  
$ drweb-ctl quarantine -a --Delete All
```



The first command will scan files in the `/home/user` directory in the autonomous copy mode. Files containing known viruses will be moved to quarantine. The second command will process quarantine content (in the autonomous copy mode as well) and remove all the objects.

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

In this section:

- [Function.](#)
- [Managing the Components.](#)
- [Threats Management.](#)
- [Managing the Settings.](#)
- [Scanning Local Files.](#)

Function

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

1. View the current state of the Dr.Web for UNIX Internet Gateways 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 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 Dr.Web for UNIX Internet Gateways to the centralized protection server or switch to the 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:

- Microsoft 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, type in the address bar of your browser the following address:

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

where *<host_with_drweb>* is the IP address or the name of the host where Dr.Web for UNIX Internet Gateways operates with the Dr.Web HTTPD web interface server, and *<port>* is the port



(on this host) which Dr.Web HTTPD is listening on. To access Dr.Web for UNIX Internet Gateways 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 address bar:

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

After the connection to the managing server is established, the startup page opens and displays the 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 Dr.Web for UNIX Internet Gateways operates.

If you need, you can provide the authorization on the web interface using a personal user certificate. To do so:

1. Create a personal certificate signed by a certificate authority certificate.
2. Import the signed certificate as a user authorization certificate in the browser that is used to connect to the web interface for management.
3. In the Dr.Web HTTPD [settings](#) (parameter `AdminSSLCA`), specify a path to the certificate authority certificate that signs your personal certificate.

If you use a personal user certificate for the authorization on the web interface, the authorization form does not appear, the user is authorized as `root`.

If necessary, refer to the [Appendix E. Generating SSL certificates](#) section.

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 on the Dr.Web for UNIX products.
- **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.



- **Log Out** ends the current web interface session (not available for authentication with user's personal certificate).

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 Dr.Web for UNIX Internet Gateways components are divided into two groups: main components, which monitor threats, and service components, which are responsible for the overall correct operation of Dr.Web for UNIX Internet Gateways. 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 distribution). For each component the following information is specified:

1. **Name.** Click the name to open the [settings page](#) containing the settings for this component;
2. **State.** The state of a component is indicated by a switch icon and by a note about the component 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 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. **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 Dr.Web for UNIX Internet Gateways components (such as the [the scan engine](#), [the file scanning component](#), and so on) 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 the **Renew** button (or the **Activate license** 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 Dr.Web for UNIX Internet Gateways 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 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, select the check boxes for the corresponding objects. To select all objects or cancel the selection, select the check box in the **File** field in the threat list 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):

	—remove (i.e. to permanently delete) selected files.
	—restore selected files from quarantine to the original location.
	—apply an additional action to the selected files (available actions are specified in the drop-down list): <ul style="list-style-type: none">• Quarantine—put the selected files that contain threats to quarantine• Cure—try to cure the threats• Ignore—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 can also switch Dr.Web for UNIX Internet Gateways into the *centralized 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 Dr.Web for UNIX Internet Gateways 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 **Centralized protection** item in the menu will take you to the [page for managing](#) the centralized protection mode.
- The **General Settings** item in the menu corresponds to the [settings](#) of the Dr.Web ConfigD component, which is responsible for the overall functioning of Dr.Web for UNIX Internet Gateways.

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 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.
- To add an additional subordinate section with settings for a component, if it is allowed, click + to the right of the component's name. Then, 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, click ✖ to the right of the subsection name (tag) that appears when you hover over the component name. Then, confirm that you want to delete the subsection and click **Yes** or close the window without deleting the subsection by clicking **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 configuration parameters that can be viewed and adjusted.
- **Changed**—show the table with the component configuration parameters that have values different from the default ones.



- **Ini Editor**—show a text editor with this component 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 the name and description of the 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 check box 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 **General**, **Advanced**, and so on). 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 name: . When a group is expanded and the parameters are displayed in the table, the following icon appears to the left of the group 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 check box). 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.



Section	Parameter	Value
General	MaxForks Maximum number of scanning processes	10
	LogLevel Logging level	Info
	Log Logging method	Auto
Advanced	FixedSocketPath Socket path	Not specified

Figure 3. Component 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 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

Cancel Save

Figure 4. Editing a list of values

After editing the value of a parameter **Save** to apply your changes and to close the window. To close the window without applying the changes click **Cancel** or click the **X** 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 text editor.

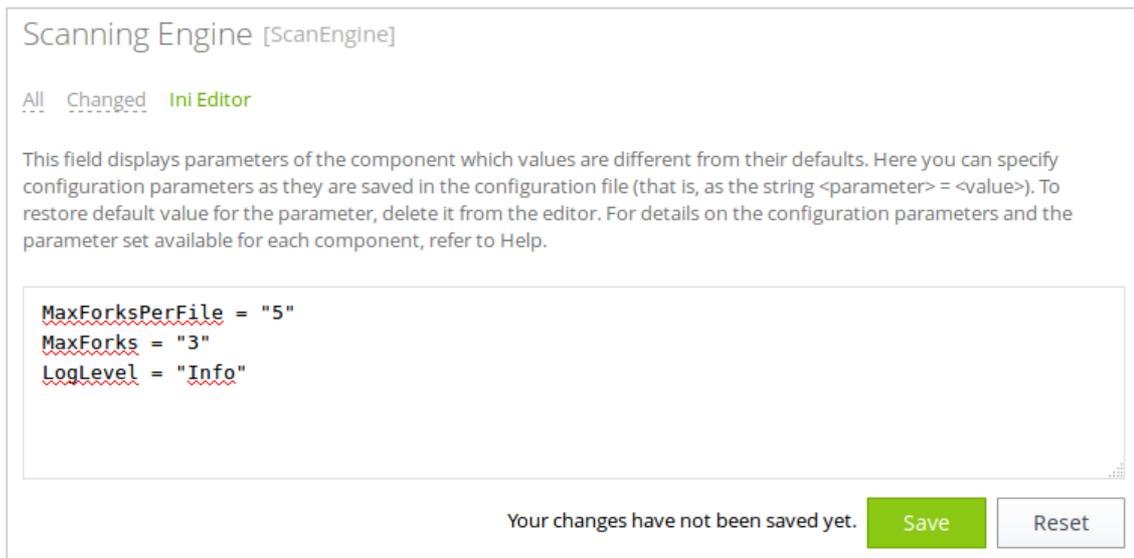


Figure 5. Built-in editor for settings

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 parameter values, click **Save** to apply the changes or click **Cancel** to cancel 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. Dr.Web for UNIX Internet Gateways 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.
- [Configuration parameters](#) of Dr.Web StatD.
- [Managing the Centralized Protection](#).

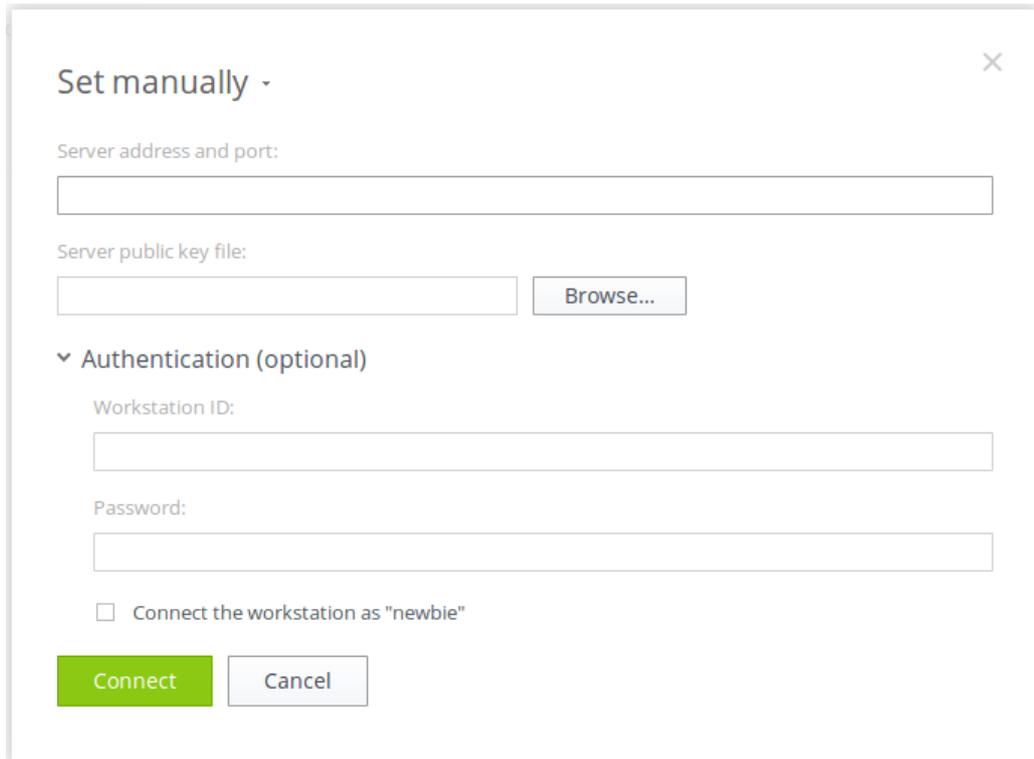
Managing the Centralized Protection

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

To connect Dr.Web for UNIX Internet Gateways to the centralized protection server or to disconnect from it, use the corresponding check box on this page.

Connection to the Centralized Protection Server

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



The screenshot shows a dialog box titled "Set manually" with a close button (X) in the top right corner. It contains the following fields and controls:

- Server address and port:** A text input field.
- Server public key file:** A text input field followed by a "Browse..." button.
- Authentication (optional):** A section header with a downward arrow.
- Workstation ID:** A text input field.
- Password:** A text input field.
- Connect the workstation as "newbie"**
- Connect** (green button) and **Cancel** (grey button) buttons at the bottom.

Figure 6. Connection to the centralized protection server

In the drop-down list located at the top of the window chose one of the methods for connecting to the centralized 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 centralized protection server. For the *Set manually* or *Detect automatically* options, you can also specify the path to the file containing the server public key (provided by your network administrator or internet service provider).

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

Moreover, you can use the **Connect the workstation as "newbie"** option (to connect as a new user). In this case, if the Newbie mode is allowed on the centralized protection server for connections from workstations, then the centralized 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 centralized 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, specify all of the parameters, click **Connect** and wait for connection to be established. To close the window without establishing a server connection, click **Cancel**.



Once you have connected Dr.Web for UNIX Internet Gateways to the centralized protection server, its operation will be managed by the centralized protection server, until you switch back to the standalone mode. Connection to the centralized 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 scan 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.



This function is available only if the Dr.Web for UNIX Internet Gateways distribution includes the Dr.Web Network Checker component.

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.

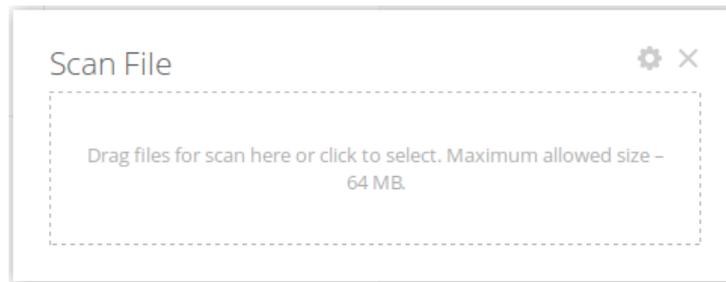


Figure 7. The scanning panel for local files

To close this panel, click **X** on the panel 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).

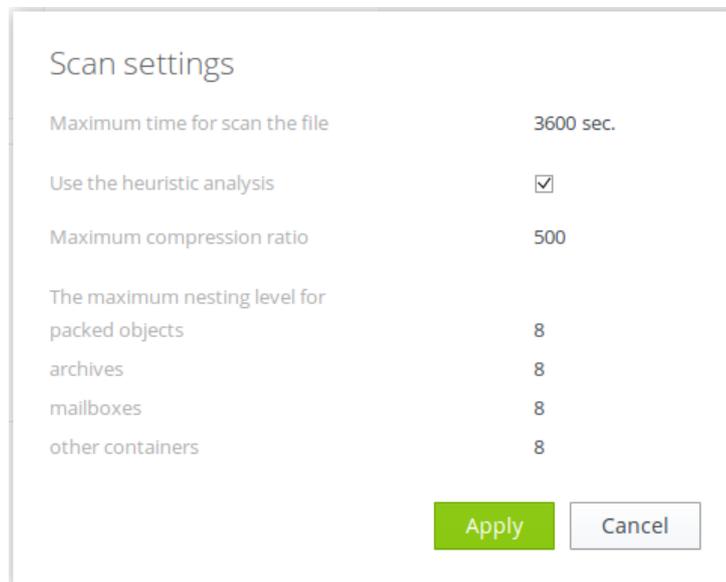


Figure 8. Setting the parameters for the scanning of local files

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 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.

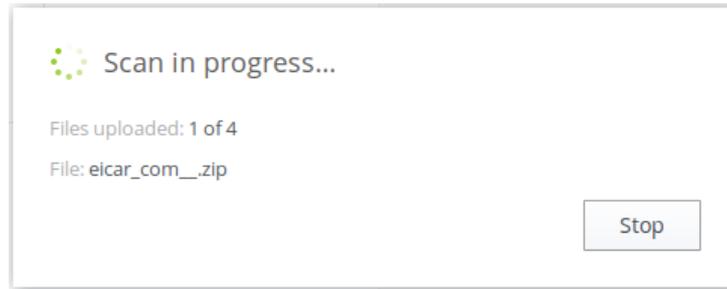


Figure 9. Current progress for the scanning of local files

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.

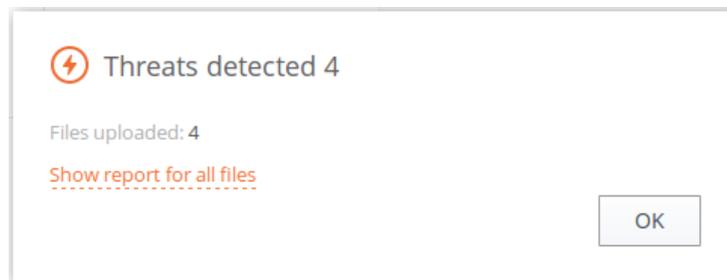


Figure 10. Results for the scanned local files

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

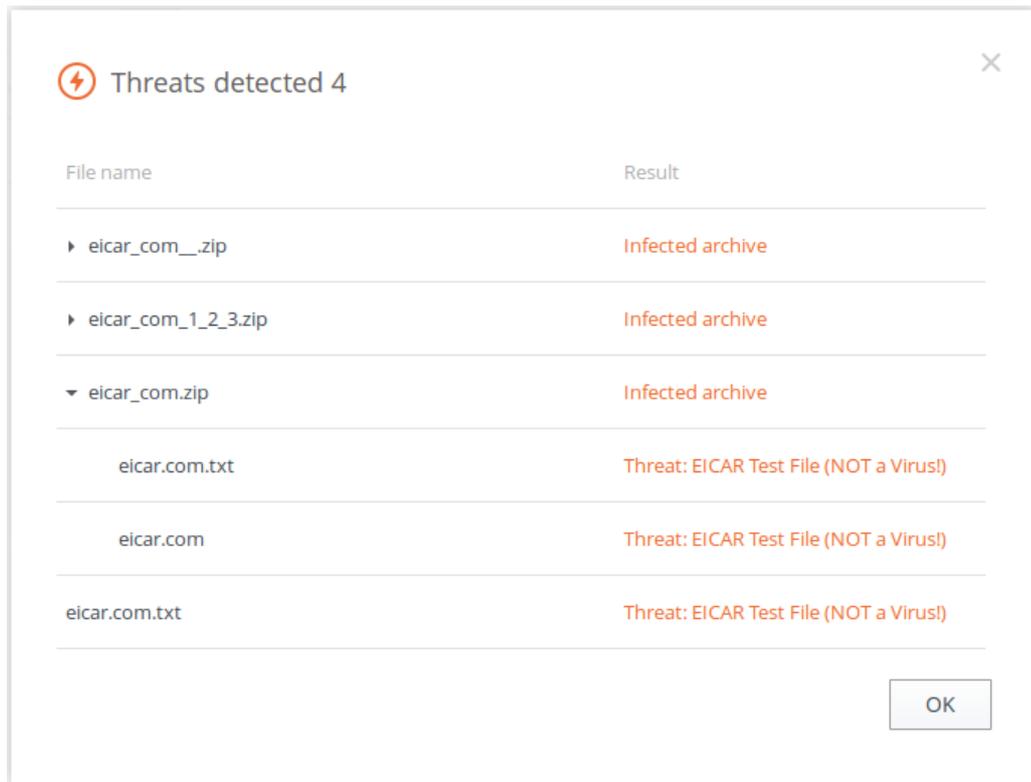


Figure 11. Extended report about the scanned local files

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.

ICAP is a lightweight HTTP-like protocol. The client sends to the ICAP server a request including headers and the encapsulated HTTP request to be checked. In the response the ICAP server returns the modified HTTP request and one or several headers

In ICAP interaction the following request types (methods) are allowed

- REQMOD—for the verification and modification of the requests;
- RESPMOD—for the verification and modification of the responses;
- OPTIONS—for obtaining the information on the connection with the ICAP server.

In REQMOD and RESPMOD requests the following headers are allowed:

- X-Client-IP—the originating IP address of the client who sent the HTTP request;
- X-Server-IP—the destination IP address of the HTTP request sent by the client
- X-Client-Username—the name of the client authenticated on the proxy server (specified as `user` or `user@domain`);

Responses to REQMOD and RESPMOD requests may include the following headers:

- X-Response-Info—is included in the response if the requested web resource belongs to a dangerous or unwanted category;



- `X-Infection-Found`—the information on virus and other potentially dangerous or suspicious objects;
- `X-Virus-ID`—the name of the detected virus;
- `X-Violations-Found`—the information of errors that occurred during the check.

The `OPTIONS` request may include the `DrWeb-Get-Scan-Status` header. Если этот заголовок имеет значение `Yes`, то в ответе будет возвращена информация об используемых версиях сканирующего движка, сканирующего ядра и вирусных баз (см. ниже). If this header has the `Yes` value, the information on the version of Dr.Web Scanning Engine, Dr.Web Core Engine and the virus bases.

The response to the `OPTIONS` request may include the following headers:

- `X-Allow-Out`—the list of header fields that Dr.Web ICAPD may include in responses;
- `X-Include`—the values of the headers retrieved from the previously received request;
- `DrWeb-Core-Engine`—the version of Dr.Web Core Engine;
- `DrWeb-Scan-Engine`—the version of Dr.Web Scan Engine;
- `DrWeb-Scan-Status`—the status of the current scanning operation
- `DrWeb-Database-Timestamp`—the virus base timestamp
- `DrWeb-Virus-Records`—the number of records in the base.

Dr.Web ICAPD may filter the web content and block access to potentially dangerous and unwanted web resources. If the user request an unwanted resource, the block page generated in accordance with the template will be returned. The page contains the information on the reason of the blocking. The block page is also returned when Dr.Web ICAPD detects an error or when an error occurs while checking the data.

To check whether a URL belongs to one of the categories, the component not only uses the database of web resource categories, which is updated regularly from the Doctor Web 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, and so on.
- *Violence*—websites that encourage violence or contain materials about various fatal accidents, and so on.
- *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, and so on.
- *Drugs*—websites that promote use, production or distribution of drugs, and so on.



- *ObsceneLanguage*—websites that contain the obscene language (in titles, articles, and so on).
- *Chats*—websites that offer a real-time transmission of text messages.
- *Terrorism*—websites that contain aggressive and propagandist materials or terroristic attacks descriptions, and so on.
- *FreeEmail*—websites that offer the possibility of free registration of an email.
- *SocialNetworks*—different social networking services: general, professional, corporate, interest-based; thematic dating sites.
- *DueToCopyrightNotice*—websites, links to which are defined by the copyright holders of some copyrighted work (movies, music, and so on).
- *OnlineGames*—websites that provide access to games using the permanent internet connection.
- *Anonymizers*—websites that allow the user to hide personal information and providing the access to the blocked web resources.
- *CryptocurrencyMiningPool*—websites that provide an access to common services for cryptocurrencies mining.
- *Jobs*—job search websites.

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 scanned for threats.

Due to special aspects of the ICAP protocol, the scanning of large portion of data (.iso images, large archives, video files, and so on) 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](#) scan 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 scan transferred data, Dr.Web ICAPD uses the [Dr.Web Network Checker](#) component. The latter one initiates scanning via the [Dr.Web Scanning Engine](#) scan engine.

To block or pass HTTP requests and responses, the Dr.Web ICAPD component can use built-in rules as well as a Lua script.



See the [Integration with Squid Proxy Server](#) section for information about integrating Dr.Web ICAPD with an HTTP proxy server.

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
<code>--help</code>	Function: Output short help information about command-line parameters to the console or to the terminal emulator and to exit upon completion. Short form: <code>-h</code> Arguments: None.
<code>--version</code>	Function: Output information about the version of this component to the console or to the terminal emulator and to exit after completion. Short form: <code>-v</code> 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 get documentation about this component of the product from the command line, use the following command: `man 1 drweb-icapd`.

Configuration Parameters

In this section

- [Component Parameters](#)
- [Rules for Traffic Monitoring and Blocking of Access](#)

The component uses configuration parameters which can be found in the [ICAPD] section of the integrated [configuration file](#) of Dr.Web for UNIX Internet Gateways.

Component Parameters

The section contains the following parameters:

Parameter	Description
LogLevel <i>{logging level}</i>	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 <i>{log type}</i>	Logging method of the component. Default value: Auto
ExePath <i>{path to file}</i>	Executable path to the component. Default value: <opt_dir>/bin/drweb-icapd. <ul style="list-style-type: none">• For GNU/Linux: /opt/drweb.com/bin/drweb-icapd.• For FreeBSD: /usr/local/libexec/drweb.com/bin/drweb-icapd
RunAsUser <i>{UID user name}</i>	The parameter determines the user on behalf of which user the component should be run. You can specify either the user's UID or login. If the login consists of numbers and cannot be distinguished from the UID, it is preceded with the "name:" prefix, for example: RunAsUser = name:123456. If the username is not specified, the component terminates with an error after the startup. Default value: drweb



Parameter	Description
Start {Boolean}	<p>Launch/do not launch the component by the Dr.Web ConfigD configuration daemon.</p> <p>When you specify the <code>Yes</code> value for this parameter, it the configuration daemon will start the component immediately; and when you specify the <code>No</code> value, the configuration daemon will terminate the component immediately.</p> <p>Default value: <code>No</code></p>
DebugDumpIcap {Boolean}	<p>Include detailed ICAP messages into the log file on the debug level (i.e. when you set <code>LogLevel = DEBUG</code>).</p> <p>Default value: <code>No</code></p>
ListenAddress {network socket}	<p>Specifies the network socket (IP address and port) on which Dr.Web ICAPD listens for connections from HTTP proxy servers.</p> <p>Default value: <code>127.0.0.1:1344</code></p>
UsePreview {Boolean}	<p>Enable/disable the <i>ICAP preview</i> mode.</p> <p><i>Do not change</i> the default value of this parameter, unless it is necessary.</p> <p>Default value: <code>Yes</code></p>
Use204 {Boolean}	<p>Return the response code <code>204</code> if <i>ICAP preview</i> mode is not enabled.</p> <p><i>Do not change</i> the default value of this parameter, unless it is necessary.</p> <p>Default value: <code>Yes</code></p>
AllowEarlyResponse {Boolean}	<p>Enable the ICAP early response mode, i.e. allow Dr.Web ICAPD to send the response to the client before the entire request has been received from the HTTP proxy server.</p> <p><i>Do not change</i> the default value of this parameter, unless it is necessary.</p> <p>Default value: <code>Yes</code></p>
TemplatesDir {path to directory}	<p>Path to the directory that contains the templates for the HTML notifications sent upon blocking a web resource.</p> <p>Default value: <code><var_dir>/templates/icapd</code>.</p> <ul style="list-style-type: none">• For GNU/Linux: <code>/var/opt/drweb.com/templates/icapd</code>.• For FreeBSD: <code>/var/drweb.com/templates/icapd</code>



Parameter	Description
<p>Whitelist</p> <p><i>{domain list}</i></p>	<p>White list of domains. The domains (as well as their subdomains) included in this list will always be accessible to users even if included in unwanted resource list.</p> <p>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).</p> <p>Example: Add to the list of domains <code>example.com</code> and <code>example.net</code>.</p> <ol style="list-style-type: none">1. Adding values to the configuration file.<ul style="list-style-type: none">• Two values per line:<pre data-bbox="778 745 1449 882">[ICAPD] Whitelist = "example.com", "example.net"</pre>• Two lines (a value per line):<pre data-bbox="778 943 1449 1079">[ICAPD] Whitelist = example.com Whitelist = example.net</pre>2. Adding values via the <code>drweb-ctl cfset</code> command:<pre data-bbox="778 1162 1449 1332"># drweb-ctl cfset ICAPD.Whitelist -a example.com # drweb-ctl cfset ICAPD.Whitelist -a example.net</pre> <div data-bbox="778 1355 1449 1877" style="background-color: #fff9c4; padding: 10px;"><p> Actual usage of the domain list indicated in this parameter depends on the <i>method</i> of its usage in the management rules of access to web sources defined for Dr.Web ICAPD.</p><p>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 <i>will be scanned for threats</i>.</p></div> <p>Default value: <i>(not set)</i></p>



Parameter	Description
Blacklist <i>{domain list}</i>	<p>Black list of domains. The domains (as well as their subdomains) included in this list will always be accessible to users even if included in unwanted resource list.</p> <p>The values in the list are 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).</p> <p>Example: Add to the list of domains <code>example.com</code> and <code>example.net</code>.</p> <ol style="list-style-type: none">Adding values to the configuration file.<ul style="list-style-type: none">Two values in a line:<pre>[ICAPD] Blacklist = "example.com", "example.net"</pre>Two lines (a value per line):<pre>[ICAPD] Blacklist = example.com Blacklist = example.net</pre>Adding values via the <code>drweb-ctl cfset</code> command:<pre># drweb-ctl cfset ICAPD.Blacklist -a example.com # drweb-ctl cfset ICAPD.Blacklist -a example.net</pre> <div style="background-color: #fff9c4; padding: 10px;"><p> Actual usage of the domain list indicated in this parameter depends on the <i>method</i> of its usage in the management rules of access to web sources defined for Dr.Web ICAPD.</p><p>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 <code>Whitelist</code> and <code>Blacklist</code>, the default rules guarantee that user access to it will be blocked.</p></div> <p>Default value: <i>(not set)</i></p>



Parameter	Description
<p>Adlist</p> <p><i>{list of strings}</i></p>	<p>A list of regular expressions that describe advertisement URLs: URL that matches any of the regular expressions listed here is considered to be an advertisement URL.</p> <p>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).</p> <p>Example: Add to the list the following expressions <code>.*ads.+</code> and <code>.*ad/.*\.gif\$</code>.</p> <ol style="list-style-type: none">1. Adding values to the configuration file.<ul style="list-style-type: none">• Two values in a line:<pre data-bbox="778 745 1449 853">[ICAPD] Adlist = ".*ads.+", ".*ad/.*\.gif\$"</pre>• Two lines (a value per line):<pre data-bbox="778 913 1449 1048">[ICAPD] Adlist = .*ads.+ Adlist = .*ad/.*\.gif\$</pre>2. Adding values via the <code>drweb-ctl cfset</code> command:<pre data-bbox="778 1115 1449 1279"># drweb-ctl cfset ICAPD.Adlist -a '.*ads.+' # drweb-ctl cfset ICAPD.Adlist -a '.*ad/.*\.gif\$'</pre> <p><i>Regular expressions</i> are specified using either the POSIX syntax (BRE, ERE) or the Perl syntax (PCRE, PCRE2).</p> <div data-bbox="778 1406 1449 1794" style="background-color: #fff9c4; padding: 10px;"><p> Actual usage of the expression list indicated in this parameter depends on the <i>method</i> of its usage in the management rules of access to web sources defined for Dr.Web ICAPD.</p><p>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 <code>Whitelist</code>.</p></div> <p>Default value: <i>(not set)</i></p>
<p>BlockInfectionSource</p> <p><i>{Boolean}</i></p>	<p>Block connections attempts to websites containing malicious software (included into the <i>InfectionSource</i> category).</p>



Parameter	Description
	<p>To activate blocking, the following rule should be added to the settings (see the details below):</p> <pre>url_category in "ICAPD.BlockCategory" : BLOCK as _match</pre> <p>Default value: Yes</p>
<code>BlockNotRecommended</code> {Boolean}	<p>Block connection attempts to non-recommended websites (included into the <i>NotRecommended</i> category).</p> <p>To activate blocking, the following rule should be added to the settings (see the details below):</p> <pre>url_category in "ICAPD.BlockCategory" : BLOCK as _match</pre> <p>Default value: Yes</p>
<code>BlockAdultContent</code> {Boolean}	<p>Block connection attempts to websites containing adult content (included into the <i>AdultContent</i> category).</p> <p>To activate blocking, the following rule should be added to the settings (see the details below):</p> <pre>url_category in "ICAPD.BlockCategory" : BLOCK as _match</pre> <p>Default value: No</p>
<code>BlockViolence</code> {Boolean}	<p>Block connection attempts to websites containing graphic violence (included into the <i>Violence</i> category).</p> <p>To activate blocking, the following rule should be added to the settings (see the details below):</p> <pre>url_category in "ICAPD.BlockCategory" : BLOCK as _match</pre> <p>Default value: No</p>
<code>BlockWeapons</code> {Boolean}	<p>Block connection attempts to websites dedicated to weapons (included into the <i>Weapons</i> category).</p> <p>To activate blocking, add the following rule to the settings (see the details below):</p> <pre>url_category in "ICAPD.BlockCategory" : BLOCK as _match</pre> <p>Default value: No</p>



Parameter	Description
BlockGambling {Boolean}	<p>Block connection attempts to gambling websites (included into the <i>Gambling</i> category).</p> <p>To activate blocking, add the following rule to the settings (see the details below):</p> <pre>url_category in "ICAPD.BlockCategory" : BLOCK as _match</pre> <p>Default value: No</p>
BlockDrugs {Boolean}	<p>Block connection attempts to websites dedicated to drugs (included into the <i>Drugs</i> category).</p> <p>To activate blocking, add the following rule to the settings (see the details below):</p> <pre>url_category in "ICAPD.BlockCategory" : BLOCK as _match</pre> <p>Default value: No</p>
BlockObsceneLanguage {Boolean}	<p>Block connection attempts to websites containing obscene language (included into the <i>ObsceneLanguage</i> category).</p> <p>To activate blocking, add the following rule to the settings (see the details below):</p> <pre>url_category in "ICAPD.BlockCategory" : BLOCK as _match</pre> <p>Default value: No</p>
BlockChats {Boolean}	<p>Block connection attempts to chat websites (included into the <i>Chats</i> category).</p> <p>To activate blocking, add the following rule to the settings (see the details below):</p> <pre>url_category in "ICAPD.BlockCategory" : BLOCK as _match</pre> <p>Default value: No</p>
BlockTerrorism {Boolean}	<p>Block connection attempts to websites dedicated to terrorism (included into the <i>Terrorism</i> category).</p> <p>To activate blocking, add the following rule to the settings (see the details below):</p>



Parameter	Description
	<pre>url_category in "ICAPD.BlockCategory" : BLOCK as _match</pre> <p>Default value: No</p>
BlockFreeEmail <i>{Boolean}</i>	<p>Block connection attempts to websites of free email services (included into the <i>FreeEmail</i> category).</p> <p>To activate blocking, add the following rule to the settings (see the details below):</p> <pre>url_category in "ICAPD.BlockCategory" : BLOCK as _match</pre> <p>Default value: No</p>
BlockSocialNetworks <i>{Boolean}</i>	<p>Block connection attempts to social networking websites (included into the <i>SocialNetworks</i> category).</p> <p>To enable blocking, the following rule should be added to the settings (see the details below):</p> <pre>url_category in "ICAPD.BlockCategory" : BLOCK as _match</pre> <p>Default value: No</p>
BlockDueToCopyrightNotice <i>{Boolean}</i>	<p>Block connection attempts to websites that were added according to copyright holder requests (included into the <i>DueToCopyrightNotice</i> category).</p> <p>To enable blocking, add the following rule to the settings (see the details below):</p> <pre>url_category in "ICAPD.BlockCategory" : BLOCK as _match</pre> <p>Default value: Yes</p>
BlockOnlineGames <i>{Boolean}</i>	<p>Block connection attempts to Online games websites (included into <i>OnlineGames</i> category).</p> <p>To enable blocking, add the following rule to the settings (see the details below):</p> <pre>url_category in "ICAPD.BlockCategory" : BLOCK as _match</pre> <p>Default value: Yes</p>
BlockAnonymizers	<p>Block connection attempts to anonymizers (included into <i>Anonymizers</i> category).</p>



Parameter	Description
<i>{Boolean}</i>	<p>To enable blocking, add the following rule to the settings (see the details below):</p> <pre>url_category in "ICAPD.BlockCategory" : BLOCK as _match</pre> <p>Default value: <i>Yes</i></p>
<code>BlockCryptocurrencyMiningPools</code> <i>{Boolean}</i>	<p>Block connection attempts to websites providing an access to common services for cryptocurrencies mining (included into <i>CryptocurrencyMiningPool</i> category).</p> <p>To enable blocking, add the following rule to the settings (see the details below):</p> <pre>url_category in "ICAPD.BlockCategory" : BLOCK as _match</pre> <p>Default value: <i>Yes</i></p>
<code>BlockJobs</code> <i>{Boolean}</i>	<p>Block connection attempts to job search websites (included into <i>Jobs</i> category).</p> <p>To enable blocking, add the following rule to the settings (see the details below):</p> <pre>url_category in "ICAPD.BlockCategory" : BLOCK as _match</pre> <p>Default value: <i>No</i></p>
<code>ScanTimeout</code> <i>{time interval}</i>	<p>Time-out for scanning one file initiated by Dr.Web ICAPD.</p> <p>Acceptable values: from 1 second (1s) to 1 hour (1h).</p> <p>Default value: <i>30s</i></p>
<code>HeuristicAnalysis</code> <i>{On Off}</i>	<p>Enable/disable heuristic analysis for detection of unknown threats during. The use of heuristic analysis raises the level of protection, but increases the duration of scanning.</p> <p>Action applied to threats detected by the heuristic analyzer is specified as the <code>BlockSuspicious</code> parameter value.</p> <p>Allowed values:</p> <ul style="list-style-type: none">• <i>On</i>—enable heuristic analysis;• <i>Off</i>—disable heuristic analysis. <p>Default value: <i>On</i></p>
<code>PackerMaxLevel</code> <i>{integer}</i>	<p>Maximum nesting level for packed objects. A packed object is executable code compressed with special software (UPX,</p>



Parameter	Description
	<p>PELock, PEXcompact, Petite, ASPack, Morphine and so on). Such objects may include other packed objects which may also include packed objects. etc. The value of this parameter specifies the nesting limit beyond which packed objects inside other packed objects will not be scanned.</p> <p>The nesting level is not limited. If the value is set to 0, nested objects are not scanned.</p> <p>Default value: 8</p>
<code>ArchiveMaxLevel</code> <i>{integer}</i>	<p>Maximum nesting level for archives (zip, rar, and so on) in which other archives may be enclosed (and these archives may also include other archives, and so on). The value of this parameter specifies the nesting limit beyond which archives enclosed in other archives will not be scanned.</p> <p>The nesting level is not limited. If the value is set to 0, nested objects are not scanned.</p> <p>Default value: 0</p>
<code>MailMaxLevel</code> <i>{integer}</i>	<p>Maximum nesting level for files of mailers (pst, tbb and so on) in which other files may be enclosed (and these files may also include other files and so on). The value of this parameter specifies the nesting limit beyond which objects inside other objects will not be scanned.</p> <p>The nesting level is not limited. If the value is set to 0, nested objects are not scanned.</p> <p>Default value: 0</p>
<code>ContainerMaxLevel</code> <i>{integer}</i>	<p>Maximum nesting level for other types of objects inside which other objects are enclosed (HTML pages, jar-files, etc.). The value of this parameter specifies the nesting limit beyond which objects inside other objects will not be scanned.</p> <p>The nesting level is not limited. If the value is set to 0, nested objects are not scanned.</p> <p>Default value: 8</p>
<code>MaxCompressionRatio</code> <i>{integer}</i>	<p>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.</p> <p>The compression ratio must not be smaller than 2.</p> <p>Default value: 500</p>



Parameter	Description
BlockKnownVirus <i>{Boolean}</i>	<p>Block either incoming or outgoing data containing known threats.</p> <p>To enable blocking, add the following rule to the settings (see the details below):</p> <pre>threat_category in "ICAPD.BlockThreat" : BLOCK as _match</pre> <p>Default value: Yes</p>
BlockSuspicious <i>{Boolean}</i>	<p>Block either incoming or outgoing data containing unknown threats (detected by the heuristic analyzer).</p> <p>To enable blocking, add the following rule to the settings (see the details below):</p> <pre>threat_category in "ICAPD.BlockThreat" : BLOCK as _match</pre> <p>Default value: Yes</p>
BlockAdware <i>{Boolean}</i>	<p>Block either incoming or outgoing data containing adware.</p> <p>To enable blocking, add the following rule to the settings (see the details below):</p> <pre>threat_category in "ICAPD.BlockThreat" : BLOCK as _match</pre> <p>Default value: Yes</p>
BlockDialers <i>{Boolean}</i>	<p>Block either incoming or outgoing data containing dialer programs.</p> <p>To enable blocking, add the following rule to the settings (see the details below):</p> <pre>threat_category in "ICAPD.BlockThreat" : BLOCK as _match</pre> <p>Default value: Yes</p>
BlockJokes <i>{Boolean}</i>	<p>Block either incoming or outgoing data containing joke programs.</p> <p>To enable blocking, add the following rule to the settings (see the details below):</p> <pre>threat_category in "ICAPD.BlockThreat" : BLOCK as _match</pre>



Parameter	Description
	Default value: No
BlockRiskware <i>{Boolean}</i>	<p>Block either incoming or outgoing data containing riskware.</p> <p>To enable blocking, add the following rule to the settings (see the details below):</p> <pre>threat_category in "ICAPD.BlockThreat" : BLOCK as _match</pre> <p>Default value: No</p>
BlockHacktools <i>{Boolean}</i>	<p>Block either incoming or outgoing data containing hacktools.</p> <p>To enable blocking, add the following rule to the settings (see the details below):</p> <pre>threat_category in "ICAPD.BlockThreat" : BLOCK as _match</pre> <p>Default value: No</p>
BlockUnchecked <i>{Boolean}</i>	<p>Block either incoming or outgoing data that cannot be scanned.</p> <p>Default value: No</p>
MessageHook <i>{path to file Lua function}</i>	<p>Script for processing HTTP messages in Lua or path to the file containing the script (see the HTTP Messages Processing in Lua section).</p> <p>If the Lua function or the file path are not specified, the messages will be processed in accordance with the rules. If the specified file is not available, the component launch will return an error.</p> <p>Default value: <i>Generated automatically</i>. For default settings, the script looks as following:</p> <pre>local dw = require "drweb" local cfg = require "drweb.config" local dwl = require "drweb.lookup" local rx = require "drweb.regex" function message_hook(ctx) if ctx.direction == "request" then local url = ctx.request.url if url.in_list(cfg.blacklist) then return "block" end if not url.in_list(cfg.whitelist) then if rx.search(cfg.adlist, url) or rx.search(cfg.adlist, url.raw) then return "block" end end end end</pre>



Parameter	Description
	<pre>end if url.in_categories(cfg.block_url_categories) then return "block" end end end if ctx.body.has_threat{category = cfg.block_threats} then return "block" end if cfg.block_unchecked and ctx.body.scan_error then return "block" end return "pass" end</pre>

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, and so on), 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 all rules are 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 the rules

For easy editing of the rules list gaps are left, i.e. `RuleSet<i>` sets that do not contain rules (`<i>`—`RuleSet` rule set number). Note that you *cannot* add the items other than `RuleSet<i>`, 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 management interface](#) (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 Dr.Web for UNIX Internet Gateways. To do that, use the `drweb-ctl reload` command.

Use the `drweb-ctl cfshow` command 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 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", log the chain of rules 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 for blocking 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 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 black list basis, further scans 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` scans the HTTP request or response for 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 IP addresses range `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 the 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
```



HTTP Messages Processing in Lua

In this section

- [General Information](#)
- [Script for Message Processing](#)
- [Tables Used in Scripts](#)
- [Available Auxiliary Modules](#)

General Information

The Dr.Web ICAPD component supports interaction via the Lua program interpreter (5.3.4 version is used; it is supplied together with Dr.Web for UNIX Internet Gateways). Scripts written in Lua can be used by the component for the analysis and processing of the HTTP protocol messages.

The analysis of a HTTP message (request or response), received for scanning from the proxy server via the ICAP protocol, is performed by means of the Lua script specified in the Dr.Web ICAPD settings as the value of the `MessageHook` parameter (it can be specified either as a snippet of Lua code or a path to the file that contains the required processing program).

Script for Message Processing

Requirements for the Script

The script must contain a global function which is an entry point to the message scanning module (Dr.Web ICAPD will call this function for processing newly received message). The processing function should match the following call conventions:

1. *Function name* is `message_hook`;
2. *The only argument* is the [MessageContext](#) table (provides access from the function to the information about the processed email message);
3. *The single return value* is a string. The return value determines the verdict for the scanned message, i.e. to skip it or to block it. Possible values:
 - `"pass"`—the message will be passed to the recipient (HTTP request to the server, HTTP response to the client).
 - `"block"`—HTTP message will not go through to the recipient, the client will receive a HTTP response with a blocked webpage.

The situation in which the function returns a different value or an error occurs when executing the function is treated as a scanning error; the response to the client depends in this case on the value of the `BlockUnchecked` configuration parameter.



Below is an example of a correct function definition, which will always return the *Pass* verdict to Dr.Web ICAPD for all HTTP messages received for scanning (the `ctx` argument is hereinafter an instance of the *MessageContext* table):

```
function message_hook(ctx)
  return "pass"
end
```

The script from the next example will block access to all resources except Dr.Web documentation website to all users except the members of the Web Admins group in Active Directory:

```
local dwl = require "drweb.lookup"

function message_hook(ctx)

  -- Not to block access to resources at the document website
  -- of Doctor Web
  if ctx.req.url.in_list{"download.geo.drweb.com"} then
    return "pass"
  end

  -- To allow access to users from the WebAdmins group
  -- in Active Directory
  if dwl.check("WebAdmins", "AD@WinRoot", ctx.icap.user) then
    return "pass"
  end

  -- Block access for all the others (to all resources)
  return "block"

end
```

Tables Used in Scripts

Table *MessageContext*

It is used as an input argument of the `message_hook` function. It provides access to the information about the processed HTTP message (its type, headers, body, information about the sender and the recipients, if available).

Field	Description	Data type
<code>direction</code>	The HTTP message type. It can have the following values: <ul style="list-style-type: none">"request"—an HTTP request."response"—an HTTP response.	String
<code>icap</code>	Information on ICAP request headers.	Table ICAP



Field	Description	Data type
request	Information on HTTP request headers.	Table Request
response	Information on HTTP response headers.	Table Response
body	Information on HTTP message body.	Table Body
Overridden metamethods: <i>None</i>		

Table ICAP

The table is used as the `icap` field of the [MessageContext](#) table. It contains the data on ICAP requests from a HTTP proxy server.

Field	Description	Data type
user	Information on user obtained from the <code>X-Client-Username</code> header of the ICAP request.	Table User
src	IP address of a client that sent the request (the address is obtained from the <code>X-Client-IP</code> header of the ICAP request sent by the proxy server), or <code>nil</code> , if the address is unknown.	Table IpAddress
field	Array of ICAP request headers.	Array of HeaderField tables
search	<p>The function for searching a header by a regular expression. It requires a <code>patterns</code> argument, i.e. search patterns, i.e. one (string) or several (array of strings) regular expressions in the Perl syntax (PCRE). It searches for all available headers. Note that when using quoted strings, the slash character must be escaped.</p> <p>Returns a Boolean value:</p> <ul style="list-style-type: none"> <code>true</code>—if the <code>field.name .. ": " .. field.value.decoded</code> string matches at least one of the specified regular expression for at least one of the headers; <code>false</code>—if no matches have been found. 	Function
value	The function that requires the only argument, the name of the header (string). It returns the value of the first header with the specified name or <code>nil</code> , if there is no header with this name.	Function
Overridden metamethods: <i>None</i>		

Table User

The table contains the name and the domain of the user; both fields are optional.



Field	Description	Data type
<code>user</code>	Username	String
<code>domain</code>	User domain.	String
Overridden metamethods:		
<ul style="list-style-type: none">• <code>__toString</code>—the function returns the <code>User</code> content as a string (in the UTF-8);• <code>__concat</code>—the function concatenates the <code>User</code> string value and another string.		

Table `HeaderField`

The table describes the HTTP or ICAP message header.

Field	Description	Data type
<code>name</code>	Header name.	String
<code>value</code>	Header value.	String
Overridden metamethods: <i>None</i>		

Table `Request`

The table describes the headers of the HTTP request.

Field	Description	Data type
<code>method</code>	HTTP protocol method used in the request (for example, "POST") or <code>nil</code> , if the ICAP request does not contain the HTTP request header.	String
<code>url</code>	URL of the resource to which the HTTP request is sent.	Table Url
<code>content_type</code>	Information from the <code>Content-Type</code> header of the HTTP request.	The ContentType table
<code>field</code>	Array of HTTP request headers.	Array of HeaderField tables
<code>search</code>	The function for searching a header by a regular expression. It requires a <code>patterns</code> argument, i.e. search patterns, i.e. one (string) or several (array of strings) regular expressions in the Perl syntax (PCRE). It searches for all available headers. Note that when using quoted strings, the slash character must be escaped. Returns a Boolean value:	Function



Field	Description	Data type
	<ul style="list-style-type: none"> • <code>true</code>—if the <code>field.name</code> .. <code>": "</code> .. <code>field.value.decoded</code> string matches at least one of the specified regular expression for at least one of the headers; • <code>false</code>—if no matches have been found. 	
<code>value</code>	The function that requires the only argument, the name of the header (string). It returns the value of the first header with the specified name or <code>nil</code> , if there is no header with this name.	Function
Overridden metamethods: <i>None</i>		

Table `ContentType`

The table describes a value obtained from the `Content-Type` header.

Field	Description	Data type
<code>type</code>	MIME type of the message part	String
<code>subtype</code>	Subtype of the message part	String
<code>param</code>	Header parameters in the form of a table array with the following fields: <ul style="list-style-type: none"> • <code>name</code> is the name of a parameter (string); • <code>value</code> is the value of a parameter (string). 	Table array
<code>match</code>	<p>The function that requires the only argument <code>media_types</code>, i.e. the array of strings describing MIME types. Each string in the list must have one of the following forms: <code>"type/subtype"</code>, <code>"type/*"</code> or <code>"*/*"</code>.</p> <p>Returns a Boolean value:</p> <ul style="list-style-type: none"> • <code>true</code>—if MIME type of the body matches to any of the specified strings (not case-sensitive) or the array contains the string <code>"*/*"</code>. • <code>false</code>—otherwise. 	Function
Overridden metamethods:		
<ul style="list-style-type: none"> • <code>__toString</code> is the function that returns the decoded header value; • <code>__concat</code> is the function that concatenates the decrypted value of the header and a string. 		

Table `Url`

The table describes a URL.



Field	Description	Data type
<code>scheme</code>	Scheme (protocol) prefix, for example, "http". If the prefix is missing, the value is <code>nil</code> .	String
<code>host</code>	Host name or IP address, for example, "example.com". If the host name is missing, <code>nil</code> .	String
<code>port</code>	Port number, for example, 80. If the number is missing, the value is <code>nil</code> .	Number
<code>path</code>	Path to a resource, for example, "index.html". If the path is missing, the value is <code>nil</code> .	String
<code>query</code>	Decoded request parameters. If the parameters are missing, the value is <code>nil</code> .	String
<code>legal_url</code>	If the URL belongs to the <code>owners_notice</code> category, the field contains a URL to the owner's website; otherwise, it is <code>nil</code> .	String
<code>in_list</code>	The function that requires the only argument— <code>hosts</code> , i.e. the host list (an array of strings). It returns a Boolean value: <ul style="list-style-type: none">• <code>true</code>—if <code>host</code> is a subdomain of one of the specified domains or matches one of them;• <code>false</code>—if <code>host</code> is not a subdomain of one of the specified domains or does not match anyone of them.	Function
<code>categories</code>	The function that receives one optional argument <code>filter</code> , i.e. the UrlCategoryFilter table (no argument is equivalent of using an empty table). It returns an iterator function using which you can iterate over all categories that meet the URL conditions specified by <code>filter</code> .	Function
<code>in_categories</code>	The function that requires the only argument— <code>categories</code> , i.e. the URL category list (an array of strings). It returns a Boolean value: <ul style="list-style-type: none">• <code>true</code>—if the URL falls under at least one of the specified categories;• <code>false</code>—if the URL does not fall under at least one of the specified categories; If the <code>categories</code> array is empty, it always returns <code>false</code> . See the possible category values in the description of the <code>category</code> field in the UrlCategoryFilter table.	Function
<code>raw</code>	A raw, undecoded URL.	Table RawUrl

Overridden metamethods:

- `__toString`—the function returns the `Url` content as a string (in the UTF-8);
- `__concat`—the function concatenates the `Url` string value and another string.



Table *RawUrl*

The table contains the undecoded URL data.

Field	Description	Data type
<code>scheme</code>	Scheme (protocol) prefix, for example, "http". If the prefix is missing, the value is, <code>nil</code> .	String
<code>host</code>	Host name or IP address, for example, "example.com". If it is missing, the value is, <code>nil</code> .	String
<code>port</code>	Port number, for example, 80. If the port number is missing, the value is <code>nil</code> .	Number
<code>path</code>	Path to a resource, for example, "index.html". If the path is missing, the value is <code>nil</code> .	String
<code>query</code>	Decoded request parameters. If the parameters are missing, the value is, <code>nil</code> .	String
Overridden metamethods: <ul style="list-style-type: none">• <code>__toString</code>—the function returns the <code>RawUrl</code> content as a string (in the UTF-8);• <code>__concat</code>—the function concatenates the <code>RawUrl</code> string value and another string.		

Table *UrlCategoryFilter*

A table describing a filter for URL categories (all fields are optional):

Field	Description	Data type
<code>category</code>	The list of categories that the URL must match (not case-sensitive). The list may contain the following values: <ul style="list-style-type: none">• "infection_source"—an infection source;• "not_recommended"—a source that is not recommended for visiting;• "adult_content"—adult content;• "violence"—violence;• "weapons"—weapons;• "gambling"—gambling;• "drugs"—drugs;• "obscene_language"—obscene language;• "chats"—chats;• "terrorism"—terrorism;• "free_email"—free email;• "social_networks"—social networks;	String or table of strings



Field	Description	Data type
	<ul style="list-style-type: none"> • "owners_notice"—websites added due to a notice from copyright owner; • "online_games"—online games; • "anonymizers"—anonymizers; • "cryptocurrency_mining_pools"—cryptocurrency mining pools; • "jobs"—job search websites; • "black_list"—black list. 	
category_not	The list of categories that the URL may not match (not case-sensitive).	String or table of strings
Overridden metamehtods: <i>None</i>		

If the filter field is not specified (the value is `nil`), any threat matches the filter.

If several filter fields are specified, the condition is combined by a conjunction (logical AND). If the filter field is a table (list), the object must match at least one of the table (list) items.

Table Response

The table describes the HTTP response headers.

Field	Description	Data type
status	HTTP response code or <code>nil</code> , if the ICAP request does not contain the HTTP response header.	Number
reason	Comment to the response code, or <code>nil</code> if the comment missing.	String
content_type	Information obtained from the <code>Content-Type</code> header of the HTTP response.	Table ContentType
field	Array of HTTP response headers.	Array of HeaderField tables
search	<p>The function for searching the header by a regular expression. Accepts a mandatory <code>patterns</code> argument, i.e. search patterns, i.e. one (a string) or several (array of strings) regular expressions in the Perl syntax (PCRE). Searches for all available headers. Note that when using quoted strings, the slash character must be escaped.</p> <p>Returns a Boolean value:</p> <ul style="list-style-type: none"> • <code>true</code>—if the <code>field.name .. ":" .. field.value.decoded</code> string matches at least one 	Function



Field	Description	Data type
	of the specified regular expression for at least one of the headers; <ul style="list-style-type: none">• <code>false</code>—if the <code>field.name .. ": " .. field.value.decoded</code> string matches does not match any regular expression for any of the headers;	
<code>value</code>	The function that requires the only argument—the header name (string). It returns the value of the first header with the specified name or <code>nil</code> , if there is no header with this name.	Function
Overridden metamehtods: <i>None</i>		

Table Body

A table describing the HTTP message body.

Field	Description	Data type
<code>has_threat</code>	The function that receives one optional argument <code>filter</code> , i.e. the ThreatFilter table (absence of argument equals using an empty table). It returns a Boolean value <ul style="list-style-type: none">• <code>true</code>—if the HTTP message body contains a threat that meets specified <code>filter</code> condition;• <code>false</code>—if there is no threat that meets the specified condition in the message body.	Function
<code>threats</code>	The function that receives one optional argument <code>filter</code> , i.e. the ThreatFilter table (absence of argument equals using an empty table). It returns an iterator function using which you can iterate over all threats detected in the HTTP message body. The threats are described using the Virus table.	Function
<code>content_type</code>	Contains the information on MIME type of the body obtained from the <code>Content-Type</code> header of the HTTP request or response (depending on what type of message is being analyzed).	Table ContentType
<code>scan_error</code>	Body scan error, if occurred, otherwise <code>nil</code> . Possible values: <ul style="list-style-type: none">• <code>"path_not_absolute"</code>—the path indicated is not absolute;• <code>"file_not_found"</code>—file was not found;• <code>"file_not_regular"</code>—the file is not a regular file;• <code>"file_not_block_device"</code>—it is not a block device;	String



Field	Description	Data type
	<ul style="list-style-type: none">• "name_too_long"—the name is too long;• "no_access"—access denied;• "read_error"—reading error occurred;• "write_error"—a writing error;• "file_too_large"—the file is too large;• "file_busy"—file is being used;• "unpacking_error"—an unpacking error;• "password_protected"—the archive is password protected;• "arch_crc_error"—CRC archive error;• "arch_invalid_header"—invalid archive header;• "arch_no_memory"—not enough memory to unpack archive;• "arch_incomplete"—incomplete archive;• "can_not_be_cured"—file cannot be cured;• "packer_level_limit"—packed object nesting level limit exceeded;• "archive_level_limit"—archive nesting level limit exceeded;• "mail_level_limit"—mail file nesting level limit exceeded;• "container_level_limit"—container nesting level limit exceeded;• "compression_limit"—compression rate limit exceeded;• "report_size_limit"—report size limit exceeded;• "scan_timeout"—scan timeout limit exceeded;• "engine_crash"—scan engine failure;• "engine_hangup"—scan engine hangup;• "engine_error"—scan engine error;• "no_license"—no active license found;• "multiscan_too_late"—multiscanning error;• "curing_limit_reached"—cure attempts limit exceeded;• "non_supported_disk"—disk type is not supported;• "unexpected_error"—an unexpected error.	
Overridden metamehtods: <i>None</i>		



Table Virus

Table that describes a threat.

Field	Description	Data type
name	Threat type (according to the Doctor Web classification)	String
type	Threat type (according to the Doctor Web classification). Possible values: <ul style="list-style-type: none">• "known_virus"—a known threat (a threat that has a description in the virus databases);• "virus_modification"—a modification of the known threat;• "unknown_virus"—an unknown threat, suspicious object;• "adware"—an advertising program;• "dialer"—a dialer program;• "joke"—a joke program;• "riskware"—a potentially dangerous program;• "hacktool"—a hacktool.	String
Overridden metamethods: <i>None</i>		

Table ThreatFilter

The table describes the filter for threats; all fields are optional.

Field	Description	Data type
category	List of categories that the threat must match (not case-sensitive). See the list of categories in description of the <code>type</code> field of the Virus table.	String or table of strings
category_not	List of categories that the threat cannot match (not case-sensitive).	String or table of strings
Overridden metamethods: <i>None</i>		

If the filter field is not specified (the value is `nil`), any threat matches the filter. If several filter fields are specified, then the condition is combined by a conjunction (logical AND). If the filter field is a table (list), the object must match at least one of the table (list) items.

Usage examples:

1. Writing to the log the names of all threats found in the HTTP message:



```
local dw = require "drweb"

function message_hook(ctx)
  for virus in ctx.body.threats() do
    dw.notice("threat found: " .. virus.name)
  end
  return "pass"
end
```

2. Writing to the log the threat names that match the category filter, and the names of the message parts where the threats have been detected:

```
local dw = require "drweb"

function message_hook(ctx)
  for v in ctx.body.threats({category = "known_virus"}) do
    dw.notice("found known virus: " .. v.name)
  end
  return "pass"
end
```

Available Auxiliary Modules

For interconnection with Dr.Web for UNIX Internet Gateways in program space in Lua the following specific modules can be imported.

Name of the module	Function
drweb	The module that provides functions to record messages from the Lua program to the log of the Dr.Web for UNIX Internet Gateways component which has launched the Lua program and the means of asynchronous execution of Lua procedures.
drweb.lookup	The module that provides tools to request data from external sources by calling the Dr.Web LookupD module.
drweb.regex	The module that provides an interface to match strings and regular expressions.
drweb.config	The module that provides a table with the Dr.Web ICAPD configuration parameter values.

Contents of the drweb Module

1. Functions

The module provides a set of functions.



- Saving messages from the Lua program in the Dr.Web for UNIX Internet Gateways component log:
 - `log(<level>, <message>)` writes the `<message>` string to the Dr.Web for UNIX Internet Gateways log on the `<level>` level (the required level is defined using the `"debug"`, `"info"`, `"notice"`, `"warning"`, and `"error"`);
 - `debug(<message>)` writes the `<message>` string to the Dr.Web for UNIX Internet Gateways log at the `DEBUG` level;
 - `info(<message>)` writes the `<message>` string to the Dr.Web for UNIX Internet Gateways log at the `INFO` level;
 - `notice(<message>)` writes the `<message>` string to the Dr.Web for UNIX Internet Gateways log at the `NOTICE` level;
 - `warning(<message>)` writes the `<message>` string to the Dr.Web for UNIX Internet Gateways log at the `WARNING` level;
 - `error(<message>)` writes the `<message>` string to the Dr.Web for UNIX Internet Gateways log at the `ERROR` level.
- Managing the synchronization of Lua procedures:
 - `sleep(<sec.>)` pauses the execution of a Lua procedure instance for a specified number of seconds.
 - `async(<Lua function> [, <argument list>])` launches the specified function asynchronously and passes to it the specified argument list. The `async` function call completes immediately, and the return value (the table `Future`) allows you to obtain the result of the `<Lua function>`.
- Adding IP addresses to the [IpAddress](#) table:
 - `ip(<address>)` indicates an IP address, sent as the `<address>` string in the form of an `IpAddress` table. Either IPv4 or IPv6 addresses can be used.
- Uploading external data from a text file:
 - `load_set(<file path>)` generates a table with the `true` values from the contents of the specified text file; strings read from a file are used as keys. Empty strings as well as strings with blank spaces will be ignored;
 - `load_array(<file path>)` generates a string array from the contents of the specified text file. Empty strings and strings consisting of whitespace characters only, are ignored and are not included in the array.

2. Tables

- The `Future` table describes the pending result of performing a function using the `async` function.

Field	Description	Data type
<code>wait</code>	A function that returns the result of the function started using the <code>async</code> function. If the function has not completed its execution yet, it waits for the completion and returns the result.	Function



Field	Description	Data type
	If the function is completed before <code>wait</code> is called, the result is returned immediately. If the started function fails, the <code>wait</code> call generates the same error.	
Overridden metamethods: <i>None</i>		

- The `IpAddress` table describes an IP address.

Field	Description	Data type
<code>belongs</code>	<p>Function checks an IP address stored in the <code>IpAddress</code> table for belonging to the specified subnets (IP address ranges).</p> <p>Receives the only argument—a string that looks like: "<code><IP address></code>" or "<code><IP address> / <mask></code>", where <code><IP address></code>—a host address or a network address (for example, "<code>127.0.0.1</code>"), and <code><mask></code>—a subnetwork mask (can be specified as an IP address, for example, "<code>255.0.0.0</code>", or in the numerical form, for example, "<code>8</code>").</p> <p>Returns a Boolean value:</p> <ul style="list-style-type: none">• <code>true</code> indicates that the address equals to at least one of the specified addresses or belongs at least one of the specified subnets (range of IP addresses);• <code>false</code>—otherwise.	Function
Overridden metamethods:		
<ul style="list-style-type: none">• <code>__toString</code> is a function that modifies <code>IpAddress</code> in a string, for example: "<code>127.0.0.1</code>" (IPv4) or "<code>:::1</code>" (IPv6);• <code>__concat</code> is a function that performs joining <code>IpAddress</code> to a string;• <code>__eq</code> is a function that checks the equality of two <code>IpAddress</code>;• <code>__band</code>—function that allows to apply a mask, for example: <code>dw.ip('192.168.1.2') & dw.ip('255.255.254.0')</code>		

3. Examples

- Writing the messages generated by a procedure initiating asynchronously to the log:



```
local dw = require "drweb"

-- This function waits two seconds and returns a string,
-- received as an argument
function out_msg(message)
  dw.sleep(2)
  return message
end

-- "Main" function
function intercept(ctx)
  -- Output of a string at the NOTICE level to the Dr.Web for UNIX Internet
  Gateways log
  dw.notice("Intercept function started.")

  -- An asynchronous start of two copies of the out_msg function
  local f1 = dw.async(out_msg, "Hello,")
  local f2 = dw.async(out_msg, " world!")

  -- Waiting for the completion of the copies of the function
  -- out_msg and output its results to log
  -- the Dr.Web for UNIX Internet Gateways log at the DEBUG level
  dw.log("debug", f1.wait() .. f2.wait())
end
```

- Creating a scheduled procedure:

```
local dw = require "drweb"

-- Save the table Future in the future global variable in order
-- to preven the removal by the garbage collector
future = dw.async(function()
  while true do
    -- Everyday, the following message is displayed in the log
    dw.sleep(60 * 60 * 24)
    dw.notice("A brand new day began")
  end
end)
```

- Modifying an IP address represented as a string into an [IpAddress](#) table::

```
local dw = require "drweb"

local ipv4 = dw.ip("127.0.0.1")
local ipv6 = dw.ip("::1")
local mapped = dw.ip("::ffff:127.0.0.1")
```

Contents of the drweb.lookup Module

1. Functions

The module provides the following functions:

- `lookup(<request>, <parameters>)` requests data from an external storage available via the



Dr.Web LookupD module. The *<request>* argument must correspond to a section in the Dr.Web LookupD settings (the string *<type>@<tag>*). The *<parameters>* argument is optional. It describes substitutions that will be used to generate a request. The following automatically permitted markers can be used:

- *\$u*, *\$U* is automatically replaced with *user*, the user name sent by the client component;
- *\$d*, *\$D* is automatically replaced with *domain*, the domain sent by the client component.

These arguments are set as a table. Keys and values of this table must be strings. The function returns an array of strings that are results of the request;

- *check(<checked string>, <request>, <parameters>)* returns *true* if *<checked string>* is found in the external repository, available via the Dr.Web LookupD module. The arguments *<request>* and *<parameters>* are equivalent to the arguments of the *lookup* function (see above). The *<checked string>* argument is supposed to be a string or a table with the *__tostring* metamethod (i.e. that can be formatted into a string).

2. Examples

- Writing to the log list of users retrieved from the `LookupD.LDAP.users` data source:

```
local dw = require "drweb"
local dwl = require "drweb.lookup"

-- "Main" function
function intercept(ctx)
  -- Writing the string at the NOTICE level to the Dr.Web for UNIX Internet
  Gateways log
  dw.notice("Intercept function started.")

  -- Writing the request results to the Dr.Web for UNIX Internet Gateways
  log
  -- to the 'ldap@users' data source
  for _, s in ipairs(dwl.lookup("ldap@users", {user="username"})) do
    dw.notice("Result for request to 'ldap@users': " .. s)
  end
end

end
```

Contents of the `drweb.regex` Module

1. Function

The module provides the following functions:

- *search(<template>, <text> [, <flags>])* returns *true* if the *<text>* string contains a substring that matches the *<template>* regular expression. The optional *<flags>* parameter (integer) is a set of flags affecting the function behavior connected with the logical OR.
- *match(<template>, <text> [, <flags>])*—the same as *search* except that the *<template>* regular expression must match the entire *<text>* string, not only its substring.

2. Available flags

- `ignore_case` ignores text case.



3. Examples

```
local rx = require "drweb.regex"

rx.search("te.?t", "some Text") -- false
rx.search("te.?t", "some Text", rx.ignore_case) -- true

rx.match("some.+", "some Text") -- true
```

Contents of the drweb.config Module

1. Function

The module does not provide any functions.

2. Available tables

The module provides a table with the following fields:

Field	Description	Data type
whitelist	Value of the <code>Whitelist</code> configuration parameter.	String array
blacklist	Value of the <code>Blacklist</code> configuration parameter.	String array
adlist	Value of the <code>Adlist</code> configuration parameter.	String array
block_url_categories	List of blocked URL categories (based in <code>Block*</code> parameter values, set to <code>Yes</code>).	String array
block_threats	List of blocked threat categories (based in <code>Block*</code> parameter values, set to <code>Yes</code>).	String array
block_unchecked	Value of the <code>BlockUnchecked</code> configuration parameter.	Logical

Overridden metamethods: *None*

3. Example



```
local cfg = require "drweb.config"

function message_hook(ctx)

  -- Block messages containing threats
  -- from the list of threats to be blocked
  if ctx.body.has_threat{category = cfg.block_threats} then
    return "block"
  end

  -- To permit access to all other resources
  return "pass"

end
```



SpIDer Gate



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

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 the Doctor Web 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, and so on.
- *Violence*—websites that encourage violence or contain materials about various fatal accidents, and so on.
- *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, and so on.
- *Drugs*—websites that promote use, production or distribution of drugs, and so on.
- *ObsceneLanguage*—websites that contain the obscene language (in titles, articles, and so on).
- *Chats*—websites that offer a real-time transmission of text messages.
- *Terrorism*—websites that contain aggressive and propaganda materials or terroristic attacks descriptions, and so on.
- *FreeEmail*—websites that offer the possibility of free registration of an email.
- *SocialNetworks*—different social networking services: general, professional, corporate, interest-based; thematic dating sites.
- *DueToCopyrightNotice*—websites, links to which are defined by the copyright holders of some copyrighted work (movies, music, and so on).
- *OnlineGames*—websites that provide access to games using the permanent internet connection.



- *Anonymizers*—websites that allow the user to hide personal information and providing the access to the blocked web resources.
- *CryptocurrencyMiningPool*—websites that provide an access to common services for cryptocurrencies mining.
- *Jobs*—job search websites.

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 scanned 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.

Dr.Web for UNIX Internet Gateways can be used in an organization to create a “wall” between the company 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 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) 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 displayed (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 the notification that the access to requested resource is impossible and the details upon the block. The similar page is displayed and returned to the client if SpIDer Gate finds a threat that must be blocked. 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.

Within Dr.Web for UNIX Internet Gateways 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.

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](#) scan 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](#) scan 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
<code>--help</code>	Function: Output short help information about command-line parameters to the console or to the terminal emulator and to exit upon completion. Short form: <code>-h</code> Arguments: None.
<code>--version</code>	Function: Output information about the version of this component to the console or to the terminal emulator and to exit after completion. Short form: <code>-v</code> 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 get 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 can be found in the `[GateD]` section of the integrated [configuration file](#) of Dr.Web for UNIX Internet Gateways.

The section contains the following parameters:

Parameter	Description
<code>LogLevel</code> <i>{logging level}</i>	<p>Logging level of the component.</p> <p>If the parameter value is not specified, the <code>DefaultLogLevel</code> parameter value from the <code>[Root]</code> section is used.</p> <p>Default value: <code>Notice</code></p>
<code>Log</code> <i>{log type}</i>	<p>Logging method of the component.</p> <p>Default value: <code>Auto</code></p>
<code>ExePath</code> <i>{path to file}</i>	<p>Executable path to the component.</p> <p>Default value: <code><opt_dir>/bin/drweb-gated</code>.</p> <ul style="list-style-type: none">• For GNU/Linux: <code>/opt/drweb.com/bin/drweb-gated</code>.• For FreeBSD: <code>/usr/local/libexec/drweb.com/bin/drweb-gated</code>
<code>RunAsUser</code> <i>{UID user name}</i>	<p>The name of the user on whose behalf the component is 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: <code>RunAsUser = name:123456</code>.</p> <p>When the username is not specified, the component operation terminates with an error after the startup.</p> <p>Default value: <code>drweb</code></p>



Parameter	Description
<code>IdleTimeLimit</code> <i>{time interval}</i>	<p>Maximum idle time for the component. When the specified period of time expires, the component shuts down.</p> <p>Acceptable values: from 10 seconds (10s) to 30 days (30d) inclusive. If the <code>None</code> value is set, the component will functionate eternally; the <code>SIGTERM</code> signal will not be sent if the components goes idle.</p> <p>Default value: 30s</p>
<code>TemplatesDir</code> <i>{path to directory}</i>	<p>Path to the directory that contains the templates for the HTML notifications sent upon blocking a web resource.</p> <p>Default value: <code><var_dir>/templates/gated</code>.</p> <ul style="list-style-type: none">• For GNU/Linux: <code>/var/opt/drweb.com/templates/gated</code>.• For FreeBSD: <code>/var/drweb.com/templates/gated</code>
<code>CaPath</code> <i>{path}</i>	<p>Path to the directory or file with system list of trusted root certificates.</p> <p>Default value: <i>Path to the list of trusted certificates</i>. The path depends on your GNU/Linux distribution.</p> <ul style="list-style-type: none">• For Astra Linux, Debian, Linux Mint, SUSE Linux and Ubuntu, usually it is a path <code>/etc/ssl/certs/</code>.• For CentOS and Fedora—a path <code>/etc/pki/tls/certs/ca-bundle.crt</code>.• For other distributions a path can be defined through results of execution of the command <code>openssl version -d</code>.• If a command is unavailable or an OS distribution could not be identified, the value <code>/etc/ssl/certs/</code> 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 Linux is an auxiliary component. It performs the 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

In this section:

- [General Information](#).
- [The mechanism of connection interception](#).
- [The order of connection interception](#).

General Information

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, as well as transit connections. To configure bypassing or redirecting rules accurately, the component can use rules that are built-in in settings as well as script, written in Lua.

The mechanism of connection interception

To intercept connections, Dr.Web Firewall for Linux uses routing tables specified in the database of routing policies (see `man ip:ip route`, `ip rule`) and the `nf_conntrack` interface of the NetFilter system component. The intercepted connections and transmitted packets are



marked with bit marks that allow Dr.Web Firewall for Linux to redirect connections and process transmitted packets correctly on different stages of the chain in NetFilter (for details, see `man iptables`).

Actions in the iptables rules

Dr.Web Firewall for Linux uses the following actions in the `iptables` rules:

- **MARK**. This action allows Dr.Web Firewall for Linux to set a specified numeric mark to a package.
- **CONNMARK**. This action allows Dr.Web Firewall for Linux to set a specified numeric mark to a connection.
- **TPROXY**. This action allows Dr.Web Firewall for Linux to redirect packages from the `PREROUTING` NetFilter chain to the specified network socket (`<IP address>:<port>`) without changing the contents of the package. The use of this action allows Dr.Web Firewall for Linux to determine the initial destination address of the connection.
- **NFQUEUE**. This action allows to send the package from the network stack of the engine to a process that operates outside the kernel space for scanning. Thus, Dr.Web Firewall for Linux connects to the queue `NFQUEUE` with the specified number via a special *Netlink* socket and gets packages that are necessary to make verdicts on processing (Dr.Web Firewall for Linux must tell NetFilter one the following verdicts: `DROP`, `ACCEPT`, `REPEAT`).

Marks of packages and connections

To mark packets, Dr.Web Firewall for Linux uses the following three out of the available 32 bits in packet and connection marks.

- the `LDM` bit (*Local Delivery Mark*). Packets with this bit in the mark are sent to the local host using the set routing rules.
- the `CPM` bit (*Client Packets Mark*). Indicates the connection between a client (a connection initiator) and a proxy, i.e. Dr.Web Firewall for Linux.
- the `SPM` bit (*Server Packets Mark*). Indicates the connection between a proxy, i.e. Dr.Web Firewall for Linux, and a server (a connection recipient).

The `LDM`, `CPM`, and `SPM` bits can be any *various* bits, not used for marking packets by other applications that perform routing connections. By default, Dr.Web Firewall for Linux chooses the appropriate (not used by other applications) bits automatically.

Routes and routing policies (ip rule, ip route)

For correct operation Dr.Web Firewall for Linux (in any of the connection scanning modes) the `ip rule` routing policy that uses the routing table with number 100 must be installed in the system:

```
from all fwmark <LDM>/<LDM> lookup 100
```

The following route must should be added to the table:



```
local default dev lo scope host
```

This routing policy guarantees that packets with the LDM bit in their marks are always sent to the local host.



Hereinafter the `<XXX>` string for the `xxx` bit is a *hexadecimal* value that equals 2^N , where N is a ordinal number of the `xxx` bit in the packet mark. For example, if the smallest (zero) bit was selected as an LDM bit, then `<LDM> = 20 = 0x1`.

NetFilter (iptables) rules

For the correct operation of Dr.Web Firewall for Linux (in any connection scanning mode), the following six rules (displayed in the `iptables-save` output command format) must be present in the `nat` and `mangle` tables of the appropriate chains of the NetFilter system component:

```
*nat

-A POSTROUTING -o lo -m comment --comment drweb-firewall -m mark --mark
<LDM>/<LDM> -j ACCEPT

*mangle

-A PREROUTING -m comment --comment drweb-firewall -m mark --mark
0x0/<CPM+SPM> -m connmark --mark <SPM>/<CPM+SPM> -j MARK --set-xmark
<LDM>/<LDM>
-A PREROUTING -p tcp -m comment --comment drweb-firewall -m mark ! --mark
<CPM+SPM>/<CPM+SPM> -m connmark --mark <CPM>/<CPM+SPM> -j TPROXY --on-port
<port> --on-ip <IP-address> --tproxy-mark <LDM>/<LDM>
-A OUTPUT -m comment --comment drweb-firewall -m mark --mark
<CPM>/<CPM+SPM> -j CONNMARK --set-xmark <CPM>/0xffffffff
-A OUTPUT -m comment --comment drweb-firewall -m mark --mark <SPM>/<CPM+SPM>
-j CONNMARK --set-xmark <SPM>/0xffffffff
-A OUTPUT -m comment --comment drweb-firewall -m mark --mark 0x0/<CPM+SPM> -
m connmark ! --mark 0x0/<CPM+SPM> -j MARK --set-xmark <LDM>/<LDM>
```



In the description below, numbers 0–5 are assigned to these rules (in the order they are listed in the document). The expression `<X+Y>` means the bit number “OR” (the sum) of the respective numbers `X` and `Y`.

Parameters `<IP address>` and `<port>` in the rule number 2 indicate the network socket where Dr.Web Firewall for Linux controls intercepted connections.

Besides, the following additional rules must be present in the `mangle` tables of the corresponding chains (`OUTPUT`, `INPUT`, `FORWARD`) when enabling the interception connection mode (outgoing, incoming, and transit) in the Dr.Web Firewall for Linux settings.

- To intercept *outgoing* (*output*) connections:



```
-A OUTPUT -p tcp -m comment --comment drweb-firewall -m tcp --tcp-flags SYN,ACK SYN -m mark --mark 0x0/<CPM+SPM> -j NFQUEUE --queue-num <ONum> --queue-bypass
```

- To intercept *incoming (input)* connections:

```
-A INPUT -p tcp -m comment --comment drweb-firewall -m tcp --tcp-flags SYN,ACK SYN -m mark --mark 0x0/<CPM+SPM> -j NFQUEUE --queue-num <INum> --queue-bypass
```

- To intercept *transit (forward)* connections:

```
-A FORWARD -p tcp -m comment --comment drweb-firewall -m tcp --tcp-flags SYN,ACK SYN -m mark --mark 0x0/<CPM+SPM> -j NFQUEUE --queue-num <FNum> --queue-bypass
```



In the description below, numbers 6, 7, and 8 are assigned to these rules (in the order they are listed in the document).

Here *<ONum>*, *<INum>* and *<FNum>* are the numbers of queues in *NFQUEUE* where Dr.Web Firewall for Linux is waiting for packages that indicate the installation of the corresponding connections (these packages with a set *SYN* flag, but without the *ACK* flag).

The order of connection interception

According to any of rules 6, 7, and 8, packets indicating a new network connection of the corresponding direction, if not marked by both *CPM* and *SPM* bits, are put in the corresponding *NFQUEUE* queue by NetFilter, where they will be read by Dr.Web Firewall for Linux via the *nf_conntrack* interface. Rules 3 and 4 mark the connection as intercepted, that is, a bit indicating the connection direction is set in the connection mark; this bit number coincides with the bit number in the packet mark. As a result, according to rules 1, 2, and 5, packets sent via this connection will be delivered by Dr.Web Firewall for Linux. Rule 0 is added on the top of the *POSTROUTING* chain in the *nat* table, so that if NAT is configured, addresses for marked packets are not transmitted (because it will interfere with the Dr.Web Firewall for Linux logic of interception and connection processing).

When a packet appears in one of the *NFQUEUE* queues, Dr.Web Firewall for Linux performs basic processing of the packet, in case incorrect rules are set in NetFilter. Then Dr.Web Firewall for Linux attempts to connect with the server using its own name and the socket marked *PSC* in accordance with rule 4. Rule 5 for local delivery will not apply, because the packet is marked with *SPM* and this rule is only applicable to packets marked with *<CPM+SPM>*.

- *If the connection to the server fails*, Dr.Web Firewall for Linux generates a client packet with the *RST* bit, replacing the pair *<IP address>:<port>* with the address of a network socket of the requested server. The *DROP* verdict is sent to *NFQUEUE* as well. The socket, used for sending the packet with the *RST* bit, is marked as *<CPM+SPM>*, so that none of the above rules will apply, so that the packet will be delivered to the client according to the usual routing rules.



- If an attempt to connect to a remote server is successful, Dr.Web Firewall for Linux copies the intercepted SYN packet and resends it from the socket marked as <LDM+CPM> to redirect the packet to the local network socket. Thanks to the set LDM bit and according to the specified routing rules, when selecting the output interface, the packet will be added to the loopback interface, and then to the NetFilter PREROUTING chain, where rule 2 will be applied. Thus, the packet will be redirected to the network socket Dr.Web Firewall for Linux unchanged. This feature allows Dr.Web Firewall for Linux to save all four elements of the connection address (IP address and port of the packet sender, IP address and port of the packet recipient).

For the network socket, where Dr.Web Firewall for Linux receives the intercepted connections according to rule 2, IP_TRANSPARENT option and the <LDM+CPM> mark are set, preventing packets sent by Dr.Web Firewall for Linux from this socket from falling into the NFQUEUE queues. When a client connects, search is performed for a paired socket, using the saved four-element address (IP address and port of the packet sender, IP address and port of the packet recipient). When the connection with the client and the server is established, it is scanned in accordance with a procedure in Lua, as well as scanning rules, specified in the Dr.Web Firewall for Linux settings. If the scans are successful and the connection is stable, the associated socket pair that connects the client and server sides is transferred to the SpIDer Gate component for analysis on the transferred data. The following interaction between clients and servers is established via a mediator, SpIDer Gate. In addition to the socket pair, associated with the client and server sides, Dr.Web Firewall for Linux sends SpIDer Gate the parameters and rules for scanning the established connection.

Simplified diagram of Dr.Web Firewall for Linux operation is given below.

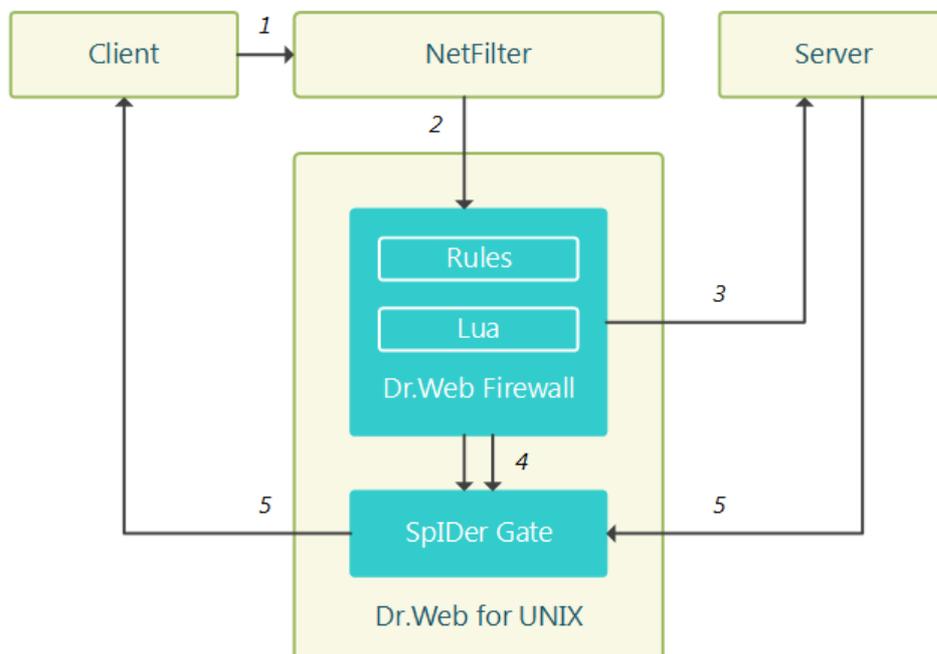


Figure 12. Diagram of the components' operation

The following steps for connection processing are marked by numbers:

1. Client attempts to connect to the server.



2. Redirecting NetFilter of the connection to Dr.Web Firewall for Linux according to the routing rules.
3. Attempt of Dr.Web Firewall for Linux to connect to the server using the client's name and connection scanning.
4. Transmission of socket pair associated with the client and server side of the connection, SpIDer Gate for connection processing, and settings and rules for the scanning.
5. Data exchange between the server and the client via SpIDer Gate as a mediator.



For the correct operation of Dr.Web Firewall for Linux, these rules in routing tables with correct numbers of bit marks, the *NFQUEUE* queues and network socket addresses for connection interception are necessary. By default settings, the component performs the necessary configuration of rules automatically. If automatic configuration of connections is disabled in the settings, it is necessary to provide the required rules manually when launching the component.

Command-Line Arguments

To run Dr.Web Firewall for Linux, type the following command in the command line:

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

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

Parameter	Description
<code>--help</code>	Function: Output short help information about command-line parameters to the console or to the terminal emulator and to exit upon completion. Short form: <code>-h</code> Arguments: None.
<code>--version</code>	Function: Output information about the version of this component to the console or to the terminal emulator and to exit after completion. Short form: <code>-v</code> Arguments: None.

Example:

```
$ /opt/drweb.com/bin/drweb-firewall --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 get 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 can be found in the `[LinuxFirewall]` section of the unified [configuration file](#) of Dr.Web for UNIX Internet Gateways.

- [Component Parameters](#).
- [Rules for Traffic Monitoring and Blocking of Access](#).

Component parameters

The section contains the following parameters:

Parameter	Description
<code>LogLevel</code> <i>{logging level}</i>	Logging level of the component. If the parameter value is not specified, the <code>DefaultLogLevel</code> parameter value from the <code>[Root]</code> section is used. Default value: <code>Notice</code>
<code>Log</code> <i>{log type}</i>	Logging method of the component. Default value: <code>Auto</code>
<code>ExePath</code> <i>{path to file}</i>	Executable path to the component. Default value: <code><opt_dir>/bin/drweb-firewall</code> . <ul style="list-style-type: none">• For GNU/Linux: <code>/opt/drweb.com/bin/drweb-firewall</code>.• For FreeBSD: <code>/usr/local/libexec/drweb.com/bin/drweb-firewall</code>
<code>XtablesLockPath</code> <i>{path to file}</i>	Path to the <code>iptables</code> (NetFilter) table blocking file. If the parameter value is not specified, the <code>/run/xtables.lock</code> and <code>/var/run/xtables.lock</code> paths are checked. If the file is not found in the specified path or default paths, when launching the component, an error occurs.



Parameter	Description
	Default value: <i>(not specified)</i>
<code>InspectFtp</code> {On Off}	Scan the data transferred over the FTP protocol. The data is scanned in accordance with the rules (see below). Default value: On
<code>InspectHttp</code> {On Off}	Scan the data transferred over the HTTP protocol. The data is scanned in accordance with the rules (see below). Default value: On
<code>InspectSmtP</code> {On Off}	Scan data transferred over SMTP protocol (if installed, the Dr.Web MailD component is used). Real data scanning will be performed according to specified scan rules (see below). Default value: On
<code>InspectPop3</code> {On Off}	Scan data transferred over POP3 protocol (if installed, the Dr.Web MailD component is used). Real data scanning will be performed according to specified scan rules (see below). Default value: On
<code>InspectImap</code> {On Off}	Scan data transferred over IMAP protocol (if installed, the Dr.Web MailD component is used). Real data scanning will be performed according to specified scan rules (see below). Default value: On
<code>AutoconfigureIptables</code> {Yes No}	Rules for configuring the NetFilter system component via the <code>iptables</code> interface. Allowed values: <ul style="list-style-type: none">• Yes—configure rules for NetFilter when launching the component and remove them when finishing its operation automatically (<i>recommended</i>);• No—do not configure the rules automatically. The required rules must be added manually by the administrator before launching the component and removed after finishing its operation.



Parameter	Description
	<div data-bbox="828 271 1449 495" style="background-color: #fff9c4; padding: 10px;"> If the automatic configuration of the rules for <code>iptables</code> is not allowed, it is necessary that the required rules for <code>iptables</code> are available before the component operation starts.</div> <p data-bbox="788 521 1010 551">Default value: <code>Yes</code></p>
<code>AutoconfigureRouting</code> <i>{Yes No}</i>	<p data-bbox="788 577 1425 645">The configuration mode for <code>ip route</code> and <code>ip rule</code> routing rules and policies.</p> <p data-bbox="788 680 970 710">Allowed values:</p> <ul data-bbox="788 736 1417 1021" style="list-style-type: none">• <code>Yes</code>—configure routing rules and policies for <code>ip route</code> and <code>ip rule</code> when launching the component and remove them when finishing its operation automatically (<i>recommended</i>);• <code>No</code>—do not perform configure rules automatically. Required rules are added manually by the administrator before launching the component and removed after finishing its operation. <div data-bbox="828 1043 1449 1301" style="background-color: #fff9c4; padding: 10px;"> If the automatic configuration of the routing rules and policies is not allowed, it is necessary that the required rules for ip route and <code>ip rule</code> are available before the component operation starts.</div> <p data-bbox="788 1328 1010 1357">Default value: <code>Yes</code></p>
<code>LocalDeliveryMark</code> <i>{integer Auto}</i>	<p data-bbox="788 1384 1445 1518">The <code><LDM></code> mark for packets that are redirected to the Dr.Web Firewall for Linux network socket (specified in the <code>TproxyListenAddress</code> parameter, see below) to intercept the connection.</p> <p data-bbox="788 1554 970 1583">Allowed values:</p> <ul data-bbox="788 1610 1417 1827" style="list-style-type: none">• <code><integer></code>—<code><LDM></code> mark for packets. Equals 2^N, where N is an <code>LDM</code> bit number in the packet, $0 \leq N \leq 31$;• <code>Auto</code>—allow Dr.Web Firewall for Linux to select the appropriate bit in the packet mark automatically (<i>recommended</i>).



Parameter	Description
	<div data-bbox="826 271 1449 824" style="background-color: #fff9c4; padding: 10px;"><p>When assigning <code><LDM></code> number manually, make sure that the corresponding bit in the packet mark is not used by any other application that manage route connections and packets (including via NetFilter). If an invalid value is specified, the component launch will fail.</p><p>Specified <code><LDM></code> number should be used in routing rules that must be added manually, if <code>AutoconfigureIptables = No</code> and/or <code>AutoconfigureRouting = No.</code></p></div> <p>Default value: <code>Auto</code></p>
<code>ClientPacketsMark</code> <i>{integer Auto}</i>	<p>The <code><CPM></code> mark for packets transferred between the client that initiates the connection and Dr.Web Firewall for Linux.</p> <p>Allowed values:</p> <ul style="list-style-type: none">• <code><integer></code>—<code><CPM></code> mark for packets. Equals 2^N, where N is a CPM bit number in the packet, $0 \leq N \leq 31$;• <code>Auto</code>—allow Dr.Web Firewall for Linux to select the appropriate bit in the packet mark automatically (<i>recommended</i>). <div data-bbox="826 1346 1449 1832" style="background-color: #fff9c4; padding: 10px;"><p>When assigning the <code><CPM></code> number manually, make sure that the corresponding bit in the packet mark is not used by any other application that manages route connections and packet (including via NetFilter). If an invalid value is specified, the component launch will fail.</p><p>Specified <code><CPM></code> number should be used in routing rules that must be added manually, if <code>AutoconfigureIptables = No.</code></p></div> <p>Default value: <code>Auto</code></p>
<code>ServerPacketsMark</code> <i>{integer Auto}</i>	<p><code><SPM></code> mark for packets transferred between Dr.Web Firewall for Linux and the server that receives the</p>



Parameter	Description
	<p>connection.</p> <p>Allowed values:</p> <ul style="list-style-type: none">• <code><integer></code>—<code><SPM></code> mark for packets. Equals 2^N, where N is a <code>SPM</code> bit number in the packet, $0 \leq N \leq 31$;• <code>Auto</code>—allow Dr.Web Firewall for Linux to select the appropriate bit in the packet mark automatically (<i>recommended</i>). <div data-bbox="828 633 1449 1189" style="background-color: #fff9c4; padding: 10px;"><p> When assigning the <code><SPM></code> number manually, make sure that the corresponding bit in the packet mark is not used by any other application that manage route connections and packets (including via NetFilter). If an invalid value is specified, the component launch will fail.</p><p>The specified <code><SPM></code> number should be used in routing rules that must be added manually, if <code>AutoconfigureIptables = No</code> and/or <code>AutoconfigureRouting = No</code>.</p></div> <p>Default value: <code>Auto</code></p>
<p><code>TproxyListenAddress</code> <i>{network socket}</i></p>	<p>Network socket (<code><IP address>:<port></code>) on which Dr.Web Firewall for Linux receives intercepted connections. If you specify port zero, it is selected automatically by the system.</p> <div data-bbox="828 1440 1449 1830" style="background-color: #fff9c4; padding: 10px;"><p> It is necessary to make sure that the corresponding socket is not used by any other application. If an invalid value is specified, the component launch will fail.</p><p>Specified IP address and port should be used in routing rules that must be added manually, if <code>AutoconfigureIptables = No</code>.</p></div> <p>Default value: <code>127.0.0.1:0</code></p>
<p><code>OutputDivertEnable</code> <i>{Yes No}</i></p>	<p>Enable/disable interception mode for incoming connections (i.e. connections initiated by applications on</p>



Parameter	Description
	<p>the remote with connections on the local host).</p> <p>Allowed values:</p> <ul style="list-style-type: none">• Yes—intercept and process outgoing connections;• No—do not intercept and process outgoing connections. <div data-bbox="828 528 1449 719" style="background-color: #fff9c4; padding: 10px;"> This setting adds or removes routing rule number 5 that is added or removed manually, if <code>AutoconfigureIptables = No</code>.</div> <p>Default value: <code>No</code></p>
<p><code>OutputDivertNfqueueNumber</code> {<i>integer</i> <i>Auto</i>}</p>	<p>Number of the queue <code>NFQUEUE</code> from which Dr.Web Firewall for Linux will retrieve SYN packages that initiate outgoing connections.</p> <p>Allowed values:</p> <ul style="list-style-type: none">• <code><integer></code>—queue number <code><ONum></code> to monitor the SYN packages of intercepted outgoing connections in <code>NFQUEUE</code>;• <code>Auto</code>—allow Dr.Web Firewall for Linux to select an appropriate queue number automatically (<i>recommended</i>). <div data-bbox="828 1229 1449 1720" style="background-color: #fff9c4; padding: 10px;"> When assigning <code><ONum></code> number manually, make sure that the corresponding queue number is not used by any other application that control connections and packages (including via the NetFilter rules). If an invalid is specified, the component launch will fail. Specified <code><ONum></code> number should be used in routing rule number 5 that is added manually, if <code>AutoconfigureIptables = No</code></div> <p>Default value: <code>Auto</code></p>
<p><code>OutputDivertConnectTransparently</code> {<i>Yes</i> <i>No</i>}</p>	<p>Enable/disable the emulation mode for connecting to the recipient (server) using the IP address of the sender of an intercepted package (client) for outgoing connections.</p>



Parameter	Description
	<p>Allowed values:</p> <ul style="list-style-type: none">• Yes—connect to the server using the address of the client that requested the connection instead of your own when intercepting the connection;• No—connect to the server from the Dr.Web Firewall for Linux address. <p>As client and Dr.Web Firewall for Linux addresses are usually the same in the outgoing connections interception mode, the default value is No.</p> <p>Default value: No</p>
<p><code>InputDivertEnable</code> {<i>Yes</i> <i>No</i>}</p>	<p>Enable/disable interception of incoming connections (i.e. connections initiated by applications on the remote host with connections on the local host).</p> <p>Allowed values:</p> <ul style="list-style-type: none">• Yes— enable intercepting and processing incoming connections;• No—disable intercepting and processing incoming connections. <div data-bbox="828 1081 1449 1339" style="background-color: #fff9c4; padding: 10px;"> This setting adds or removes routing rule number 6 that is added or removed manually, if <code>AutoconfigureIptables = No</code>. If the invalid value is specified, the component launch will fail.</div> <p>Default value: No</p>
<p><code>InputDivertNfqueueNumber</code> {<i>integer</i> <i>Auto</i>}</p>	<p>Number of the queue <code>NFQUEUE</code> from which Dr.Web Firewall for Linux will retrieve SYN packages that initiate incoming connections.</p> <p>Allowed values:</p> <ul style="list-style-type: none">• <code><integer></code>—queue number <code><INum></code> to monitor the SYN packages of intercepted outgoing connections in <code>NFQUEUE</code>;• Auto—allow Dr.Web Firewall for Linux to select an appropriate queue number automatically (<i>recommended</i>).



Parameter	Description
	<div data-bbox="826 271 1449 757" style="background-color: #fff9c4; padding: 10px;"><p>When assigning <code><INum></code> number manually, make sure that the corresponding queue number is not used by any other application that control connections and pack (including via the NetFilter rules). If an invalid is specified, the component launch will fail.</p><p>Specified <code><INum></code> number should be used in routing rule number 6 that is added manually, if <code>AutoconfigureIptables = No</code></p></div> <p>Default value: <code>Auto</code></p>
<code>InputDivertConnectTransparently</code> {Yes No}	<p>Enable/disable emulation mode for connecting to the recipient (server) using the IP address of the sender of an intercepted package (client) for incoming connections.</p> <p>Allowed values:</p> <ul style="list-style-type: none">• <code>Yes</code>—connect to the server using the address of the client that requested the connection instead of your own when intercepting the connection;• <code>No</code>—connection to the server from the Dr.Web Firewall for Linux address. <p>In the incoming connections interception mode, all traffic goes through Dr.Web Firewall for Linux, and it is possibly to connect safely to the server using the fraudulent client's addresses. This is why the default value is <code>Yes</code>.</p> <p>Default value: <code>Yes</code></p>
<code>ForwardDivertEnable</code> {Yes No}	<p>Enable/disable the interception of transit connections (i.e. connections initiated by applications on one remote host with connections on the other remote host).</p> <p>Allowed values:</p> <ul style="list-style-type: none">• <code>Yes</code>—enable intercepting and processing transit connections;• <code>No</code>—disable intercepting and processing transit connections.



Parameter	Description
	<div data-bbox="826 271 1449 456" style="background-color: #fff9c4; padding: 10px;"> This setting adds or removes routing rule number 7 that is added or removed manually, if <code>AutoconfigureIptables = No</code>.</div> <p data-bbox="788 488 995 517">Default value: <code>No</code></p>
<code>ForwardDivertNfqueueNumber</code> <i>{integer Auto}</i>	<p data-bbox="788 546 1442 640">The number of the queue <code>NFQUEUE</code> from which Dr.Web Firewall for Linux will retrieve SYN packages that initiate transit connections.</p> <p data-bbox="788 680 970 710">Allowed values:</p> <ul data-bbox="788 734 1442 949" style="list-style-type: none">• <code><integer></code>—queue number <code><FNum></code> to monitor the SYN packages of intercepted transit connections in <code>NFQUEUE</code>;• <code>Auto</code>—allow Dr.Web Firewall for Linux to select an appropriate queue number automatically (<i>recommended</i>). <div data-bbox="826 972 1449 1458" style="background-color: #fff9c4; padding: 10px;"> When assigning <code><FNum></code> number manually, make sure that the corresponding queue number is not used by any other application that control connections and pack (including via the NetFilter rules). If an invalid is specified, the component launch will fail. Specified <code><FNum></code> number should be used in routing rule number 7 that is added manually, if <code>AutoconfigureIptables = No</code></div> <p data-bbox="788 1487 1027 1516">Default value: <code>Auto</code></p>
<code>ForwardDivertConnectTransparently</code> <i>{Yes No}</i>	<p data-bbox="788 1543 1442 1637">Emulation mode for connecting to the recipient (server) using the IP address of the sender of an intercepted package (client) for transit connections.</p> <p data-bbox="788 1677 970 1706">Allowed values:</p> <ul data-bbox="788 1731 1442 1924" style="list-style-type: none">• <code>Yes</code>—connect to the server using the address of the client that requested the connection instead of your own when intercepting the connection;• <code>No</code>—connect to the server from the Dr.Web Firewall for Linux address.



Parameter	Description
	<p>As there is no guarantee that in the transit connections interception mode all the traffic goes through the same host (router) on which have Dr.Web Firewall for Linux is installed, the default value is <code>No</code> for the correct operation. If the network configuration guarantees that protected applications are use same router, the parameter can be set to <code>Yes</code>, and in this case, Dr.Web Firewall for Linux will always evaluate connection to client's addresses when connecting to servers.</p> <p>Default value: <code>No</code></p>
<code>ExcludedProc</code> <i>{path to file}</i>	<p>White list of processes (process for which the network activity is not monitored).</p> <p>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).</p> <p>Example: Add to the list of processes <code>wget</code> and <code>curl</code>.</p> <ol style="list-style-type: none">Adding values to the configuration file.<ul style="list-style-type: none">Two values in a line:<pre data-bbox="831 1111 1449 1249">[LinuxFirewall] ExcludedProc = "/usr/bin/wget", "/usr/bin/curl"</pre>Two lines (a value per line):<pre data-bbox="831 1308 1449 1447">[LinuxFirewall] ExcludedProc = /usr/bin/wget ExcludedProc = /usr/bin/curl</pre>Adding values via the <code>drweb-ctl cfset</code> command:<pre data-bbox="831 1541 1449 1774"># drweb-ctl cfset LinuxFirewall.ExcludedProc - a /usr/bin/wget # drweb-ctl cfset LinuxFirewall.ExcludedProc - a /usr/bin/curl</pre>



Parameter	Description
	<div data-bbox="828 271 1449 656" style="background-color: #fff9c4; padding: 10px;"><p>Actual usage of the process list indicated in this parameter depends on the <i>method</i> of its usage in the scanning rules defined for Dr.Web Firewall for Linux.</p><p>The list of default rules (see below) guarantees that traffic of all processes from the list is allowed <i>without any scanning</i>.</p></div> <p>Default value: <i>(not set)</i></p>
UnwrapSsl <i>{Boolean}</i>	<p>Scan encrypted traffic transferred via the SSL/TLS connections.</p> <div data-bbox="828 837 1449 1361" style="background-color: #e8f5e9; padding: 10px;"><p>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 <code>SET Unwrap_SSL = true/false</code> action (see below).</p><p>If you change the value of this parameter with the help of the <code>cfset</code> command of the <code>drweb-ctl</code> utility or with the help of the web interface, the affected dependent rules will adapt automatically.</p></div> <p>Default value: No</p>
BlockInfectionSource <i>{Boolean}</i>	<p>Block connection attempts to websites containing malicious software (included into the <i>InfectionSource</i> category).</p> <p>To activate blocking, the following rule should be added to the settings (see the details below):</p> <div data-bbox="790 1671 1449 1809" style="border: 1px solid #ccc; padding: 5px;"><pre>url_category in "LinuxFirewall.BlockCategory" : Block as _match</pre></div> <p>Default value: Yes</p>
BlockNotRecommended <i>{Boolean}</i>	<p>Block connection attempts to non-recommended websites (included into the <i>NotRecommended</i> category).</p>



Parameter	Description
	<p>To activate blocking, the following rule should be added to the settings (see the details below):</p> <pre>url_category in "LinuxFirewall.BlockCategory" : Block as _match</pre> <p>Default value: Yes</p>
BlockAdultContent <i>{Boolean}</i>	<p>Block connection attempts to websites containing adult content (included into the <i>AdultContent</i> category).</p> <p>To activate blocking, the following rule should be added to the settings (see the details below):</p> <pre>url_category in "LinuxFirewall.BlockCategory" : Block as _match</pre> <p>Default value: No</p>
BlockViolence <i>{Boolean}</i>	<p>Block connection attempts to websites containing graphic violence (included into the <i>Violence</i> category).</p> <p>To activate blocking, the following rule should be added to the settings (see the details below):</p> <pre>url_category in "LinuxFirewall.BlockCategory" : Block as _match</pre> <p>Default value: No</p>
BlockWeapons <i>{Boolean}</i>	<p>Block connection attempts to websites dedicated to weapons (included into the <i>Weapons</i> category).</p> <p>To activate blocking, the following rule should be added to the settings (see the details below):</p> <pre>url_category in "LinuxFirewall.BlockCategory" : Block as _match</pre> <p>Default value: No</p>
BlockGambling <i>{Boolean}</i>	<p>Block connection attempts to gambling websites (included into the <i>Gambling</i> category).</p> <p>To activate blocking, the following rule should be added to the settings (see the details below):</p>



Parameter	Description
	<pre>url_category in "LinuxFirewall.BlockCategory" : Block as _match</pre> <p>Default value: No</p>
BlockDrugs {Boolean}	<p>Block connection attempts to websites dedicated to drugs (included into the <i>Drugs</i> category).</p> <p>To activate blocking, the following rule should be added to the settings (see the details below):</p> <pre>url_category in "LinuxFirewall.BlockCategory" : Block as _match</pre> <p>Default value: No</p>
BlockObsceneLanguage {Boolean}	<p>Block connection attempts to websites containing obscene language (included into the <i>ObsceneLanguage</i> category).</p> <p>To activate blocking, the following rule should be added to the settings (see the details below):</p> <pre>url_category in "LinuxFirewall.BlockCategory" : Block as _match</pre> <p>Default value: No</p>
BlockChats {Boolean}	<p>Block connection attempts to chat websites (included into the <i>Chats</i> category).</p> <p>To activate blocking, the following rule should be added to the settings (see the details below):</p> <pre>url_category in "LinuxFirewall.BlockCategory" : Block as _match</pre> <p>Default value: No</p>
BlockTerrorism {Boolean}	<p>Block connection attempts to websites dedicated to terrorism (included into the <i>Terrorism</i> category).</p> <p>To activate blocking, the following rule should be added to the settings (see the details below):</p>



Parameter	Description
	<pre>url_category in "LinuxFirewall.BlockCategory" : Block as _match</pre> <p>Default value: No</p>
<code>BlockFreeEmail</code> { <i>Boolean</i> }	<p>Block connection attempts to websites of free email services (included into the <i>FreeEmail</i> category).</p> <p>To activate blocking, the following rule should be added to the settings (see the details below):</p> <pre>url_category in "LinuxFirewall.BlockCategory" : Block as _match</pre> <p>Default value: No</p>
<code>BlockSocialNetworks</code> { <i>Boolean</i> }	<p>Block connection attempts to social networking websites (included into the <i>SocialNetworks</i> category).</p> <p>To activate blocking, the following rule should be added to the settings (see the details below):</p> <pre>url_category in "LinuxFirewall.BlockCategory" : Block as _match</pre> <p>Default value: No</p>
<code>BlockDueToCopyrightNotice</code> { <i>Boolean</i> }	<p>Block connection attempts to websites that were added according to copyright holder requests (included into the <i>DueToCopyrightNotice</i> category).</p> <p>To activate blocking, the following rule should be added to the settings (see the details below):</p> <pre>url_category in "LinuxFirewall.BlockCategory" : Block as _match</pre> <p>Default value: No</p>
<code>BlockOnlineGames</code> { <i>Boolean</i> }	<p>Block connection attempts to Online games websites (included into <i>OnlineGames</i> category).</p> <p>To activate blocking, the following rule should be added to the settings (see the details below):</p>



Parameter	Description
	<pre>url_category in "LinuxFirewall.BlockCategory" : Block as _match</pre> <p>Default value: No</p>
BlockAnonymizers {Boolean}	<p>Block connection attempts to anonymizers websites (included into <i>Anonymizers</i> category).</p> <p>To activate blocking, the following rule should be added to the settings (see the details below):</p> <pre>url_category in "LinuxFirewall.BlockCategory" : Block as _match</pre> <p>Default value: No</p>
BlockCryptocurrencyMiningPools {Boolean}	<p>Block connection attempts to websites provide an access to common services for cryptocurrencies mining (included into <i>CryptocurrencyMiningPool</i> category).</p> <p>To activate blocking, the following rule should be added to the settings (see the details below):</p> <pre>url_category in "LinuxFirewall.BlockCategory" : Block as _match</pre> <p>Default value: No</p>
BlockJobs {Boolean}	<p>Block connection attempts to job search websites (included into <i>Jobs</i> category).</p> <p>To activate blocking, the following rule should be added to the settings (see the details below):</p> <pre>url_category in "LinuxFirewall.BlockCategory" : Block as _match</pre> <p>Default value: No</p>
Whitelist {domain list}	<p>White list of domains (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).</p> <p>The values in the list must be separated with commas (each value in the quotation marks). The parameter can</p>



Parameter	Description
	<p>be specified more than once in the section (in this case, all its values are combined into one list).</p> <p>Example: Add to the list of domains <code>example.com</code> and <code>example.net</code>.</p> <ol style="list-style-type: none">Adding of values to the configuration file.<ul style="list-style-type: none">Two values in one string:<pre data-bbox="831 555 1449 689">[LinuxFirewall] Whitelist = "example.com", "example.net"</pre>Two strings (one value per a string):<pre data-bbox="831 752 1449 887">[LinuxFirewall] Whitelist = example.com Whitelist = example.net</pre>Adding values via the <code>drweb-ctl cfset</code> command:<pre data-bbox="831 987 1449 1211"># drweb-ctl cfset LinuxFirewall.Whitelist -a example.com # drweb-ctl cfset LinuxFirewall.Whitelist -a example.net</pre> <div data-bbox="831 1245 1449 1966" style="background-color: #fff9c4; padding: 10px;"><p> Actual usage of the domain list indicated in this parameter depends on the <i>method</i> of its usage in the scanning rules defined for Dr.Web Firewall for Linux.</p><p>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 <i>will be scanned for threats</i> (due to the fact that data is returned in a response, and a variable direction has a value response).</p></div>



Parameter	Description
	Default value: <i>(not set)</i>
Blacklist <i>{domain list}</i>	<p>List of domains that <i>can be used as the black list</i> (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).</p> <p>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).</p> <p>Example: Add to the list of domains <code>example.com</code> and <code>example.net</code>.</p> <ol style="list-style-type: none">Adding of values to the configuration file.<ul style="list-style-type: none">Two values in one string:<pre data-bbox="831 871 1449 1010">[LinuxFirewall] Blacklist = "example.com", "example.net"</pre>Two strings (one value per a string):<pre data-bbox="831 1068 1449 1207">[LinuxFirewall] Blacklist = example.com Blacklist = example.net</pre>Adding values via the <code>drweb-ctl cfset</code> command:<pre data-bbox="831 1301 1449 1534"># drweb-ctl cfset LinuxFirewall.Blacklist -a example.com # drweb-ctl cfset LinuxFirewall.Blacklist -a example.net</pre>



Parameter	Description
	<p data-bbox="855 304 922 365"></p> <p data-bbox="970 304 1398 465">Actual usage of the domain list indicated in this parameter depends on the <i>method</i> of its usage in the scanning rules defined for Dr.Web Firewall for Linux.</p> <p data-bbox="970 506 1430 801">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 <code>Whitelist</code> and <code>Blacklist</code>, the default rules guarantee that user access to it will be blocked.</p> <p data-bbox="788 857 1043 887">Default value: <i>(not set)</i></p>
<code>ScanTimeout</code> <i>{time interval}</i>	<p data-bbox="788 913 1433 943">Time-out for scanning one file initiated by SpIDer Gate.</p> <p data-bbox="788 983 1422 1012">Acceptable values: from 1 second (1s) to 1 hour (1h).</p> <p data-bbox="788 1043 1007 1072">Default value: 30s</p>
<code>HeuristicAnalysis</code> <i>{On Off}</i>	<p data-bbox="788 1104 1445 1227">Enable/disable heuristic analysis for detection of known threats. Heuristic analysis provides higher detection reliability but, at the same time, it increases time of virus scanning.</p> <p data-bbox="788 1267 1374 1361">Action applied to threats detected by the heuristic analyzer is specified as the <code>BlockSuspicious</code> parameter value.</p> <p data-bbox="788 1402 967 1431">Allowed values:</p> <ul data-bbox="788 1458 1342 1536" style="list-style-type: none">• <code>On</code>—enable heuristic analysis when scanning;• <code>Off</code>—disable heuristic analysis. <p data-bbox="788 1559 991 1588">Default value: <code>On</code></p>
<code>PackerMaxLevel</code> <i>{integer}</i>	<p data-bbox="788 1619 1445 1877">Maximum nesting level for packed objects. A packed object is executable code compressed with special software (UPX, PELock, PECompact, Petite, ASPack, Morphine and so on). Such objects may include other packed objects which may also include packed objects, etc. The value of this parameter specifies the nesting limit beyond which packed objects inside other packed objects will not be scanned.</p> <p data-bbox="788 1917 1414 1984">The nesting level is not limited. If the value is set to 0, nested objects are not scanned.</p>



Parameter	Description
	Default value: 8
ArchiveMaxLevel <i>{integer}</i>	<p>Maximum nesting level for archives (zip, rar, and so on) in which other archives may be enclosed (and these archives may also include other archives, and so on). The value of this parameter specifies the nesting limit beyond which archives enclosed in other archives will not be scanned.</p> <p>The nesting level is not limited. If the value is set to 0, nested objects are not scanned.</p> <p>Default value: 8</p>
MailMaxLevel <i>{integer}</i>	<p>Maximum nesting level for files of mailers (pst, tbb and so on) in which other files may be enclosed (and these files may also include other files and so on). The value of this parameter specifies the nesting limit beyond which objects inside other objects will not be scanned.</p> <p>The nesting level is not limited. If the value is set to 0, nested objects are not scanned.</p> <p>Default value: 8</p>
ContainerMaxLevel <i>{integer}</i>	<p>Maximum nesting for other types objects inside which other objects are enclosed (HTML pages, jar-files, etc.). The value of this parameter specifies the nesting limit beyond which objects inside other objects will not be scanned.</p> <p>The nesting level is not limited. If the value is set to 0, nested objects are not scanned.</p> <p>Default value: 8</p>
MaxCompressionRatio <i>{integer}</i>	<p>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.</p> <p>The compression ratio must not be smaller than 2.</p> <p>Default value: 500</p>
BlockKnownVirus <i>{Boolean}</i>	<p>Block either incoming or outgoing data containing known threats.</p> <p>To activate blocking, the following rule should be added to the settings (see the details below):</p>



Parameter	Description
	<pre>threat_category in "LinuxFirewall.BlockThreat" : Block as _match</pre> <p>Default value: Yes</p>
BlockSuspicious <i>{Boolean}</i>	<p>Block either incoming or outgoing data containing unknown threats detected by the heuristic analyzer.</p> <p>To activate blocking, the following rule should be added to the settings (see the details below):</p> <pre>threat_category in "LinuxFirewall.BlockThreat" : Block as _match</pre> <p>Default value: Yes</p>
BlockAdware <i>{Boolean}</i>	<p>Block either incoming or outgoing data containing adware.</p> <p>To activate blocking, the following rule should be added to the settings (see the details below):</p> <pre>threat_category in "LinuxFirewall.BlockThreat" : Block as _match</pre> <p>Default value: Yes</p>
BlockDialers <i>{Boolean}</i>	<p>Block either incoming or outgoing data containing dialer programs.</p> <p>To activate blocking, the following rule should be added to the settings (see the details below):</p> <pre>threat_category in "LinuxFirewall.BlockThreat" : Block as _match</pre> <p>Default value: Yes</p>
BlockJokes <i>{Boolean}</i>	<p>Block either incoming or outgoing data containing joke programs.</p> <p>To activate blocking, the following rule should be added to the settings (see the details below):</p> <pre>threat_category in "LinuxFirewall.BlockThreat" : Block as _match</pre>



Parameter	Description
	Default value: No
BlockRiskware <i>{Boolean}</i>	<p>Block either incoming or outgoing data containing riskware.</p> <p>To activate blocking, the following rule should be added to the settings (see the details below):</p> <pre>threat_category in "LinuxFirewall.BlockThreat" : Block as _match</pre> <p>Default value: No</p>
BlockHacktools <i>{Boolean}</i>	<p>Block either incoming or outgoing data containing hacktools.</p> <p>To activate blocking, the following rule should be added to the settings (see the details below):</p> <pre>threat_category in "LinuxFirewall.BlockThreat" : Block as _match</pre> <p>Default value: No</p>
BlockUnchecked <i>{Boolean}</i>	<p>Block the traffic that cannot be scanned.</p> <div data-bbox="826 1182 1449 1512" style="background-color: #e6f2e6; padding: 10px;"><p>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.</p></div> <p>Default value: No</p>
InterceptHook <i>{path to file Lua function}</i>	<p>Script for processing connections in Lua or path to the file containing the script (see the Processing connections in Lua section).</p> <p>If unavailable file is specified, an error appears when loading the component.</p> <p>Default value:</p> <pre>local dwl = require 'drweb.lookup' function intercept_hook(ctx)</pre>



Parameter	Description
	<pre>-- do not check if group == Root.TrustedGroup if ctx.divert == "output" and ctx.group == "drweb" then return "pass" end -- do not check connections from privileged ports -- except FTP active mode if ctx.src.port >= 0 and ctx.src.port <= 1024 and ctx.src.port ~= 20 then return "pass" end return "check" end</pre>



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 all rules are 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 rules (`<i>`—`RuleSet` rule set number). Note that you *cannot* add the items other than `RuleSet<i>`, 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 management interface](#) (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 Dr.Web for UNIX Internet Gateways. To do that, use the `drweb-ctl reload` command.

Use the `drweb-ctl cfshow` command 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", log the chain of rules 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 for blocking 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 (Sntp), threat_category in
"LinuxFirewall.BlockThreat" : REJECT
RuleSet9 = protocol in (Sntp), url_category in "LinuxFirewall.BlockCategory"
: REJECT
RuleSet9 = protocol in (Sntp), 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 regulate the processing of the outgoing HTTP connections:

1. If the host to which a connection is established is included in a black list, the connection is blocked because the host is in the black list. Other scans are not performed.



2. If the host is included in a white list, the connection is skipped, and other scans are not performed.
3. If the URL requested by the client is in the categories of web unwanted resources, the connection is blocked due to the detection of a threat. Other scans are not performed.
4. If the response received from a remote host includes threats via HTTP contains a threat belonging to the blocked categories, the connection is blocked because the threat was detected. Other scans 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 scans are not performed.

These five rules will work only if `On` is specified in the `InspectHttp` parameter. Otherwise, none of these rules will work.

The following six rules that are specified in the `RuleSet9` control the scanning of the data sent and received via email protocols (over SMTP, POP3 or IMAP protocol); these rules are activated in the following cases:

- the transmitted email message contains attachments;
- the transmitted email message contains URLs belonging to unwanted categories;
- the transmitted email message is qualified as spam (with the spam index not less than 0.8).

If the email message 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 Anti-Spam 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 the examination of transmitted email messages for malware attachments and signs of spam. 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.



Dr.Web MailD is not included in Dr.Web for UNIX Internet Gateways.



Examples of Rules for Traffic Monitoring and Blocking of Access

1. Allow users with IP addresses in *10.10.0.0–10.10.0.254* range an HTTP access to 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 the 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`.

Processing Connections in Lua

In this section:

- [General Information.](#)
- [Requirements for the Script of Connection Processing.](#)
- [Examples.](#)
- [Tables in Use.](#)
- [Available Auxiliary Modules.](#)



General Information

Dr.Web Firewall for Linux supports interaction via program interpreter in Lua (version 5.3.4 is used and is supplied together with Dr.Web for UNIX Internet Gateways). Scripts written in Lua can be used by the component for preliminary connection scanning before it is sent to SpIDer Gate for analysis.

Connections will be analyzed with the Lua script, if the path to this script is specified in the Dr.Web Firewall for Linux settings (in the `InterceptHook` parameter). Otherwise, connection processing is performed by using the default settings and processing rules specified in the component settings (the `RuleSet*` parameters).



For more examples of Lua scripts for connection processing, follow the link: <https://github.com/DoctorWebLtd/drweb-lua-examples/tree/master/firewall>.

Requirements for the Script of Connection Processing

The script must contain a global function, which is the entry point in the connection scanning module (Dr.Web Firewall for Linux calls this function for processing a newly received connection). The function should match the following call conventions:

- *function name*—`intercept_hook`;
- *the only argument*—the Lua `context` table (provides the access to information from function on the processed connection, see description of the table below);
- *the only return value*—one of the string values from the table below:

Value	Verdict description
<code>pass</code>	Skip the connection without checking it by SpIDer Gate
<code>check</code>	Check the connection with the help of SpIDer Gate
<code>reject</code>	Discard the connection (the client who initiated the connection will receive the TCP package with an <code>RST</code> flag)
<code>drop</code>	Disconnect (the client that initiated the connection will receive no acknowledgement)

Examples

1. The script always returns the `pass` verdict (skip) for all the connections lest the connections be checked by Dr.Web Firewall for Linux:



```
-- Function of connection scanning written by the user
function intercept_hook(ctx)
  return "pass" -- do not scan the connection
end
```

2. With the help of the script given below Dr.Web Firewall for Linux checks all the connections being established with the following exceptions:

- outgoing local connections from applications running with user rights from the `drweb` group;
- connections initiated from privileged ports (regardless of the connection owner and its direction);
- connections originating from IP addresses of the local network:

```
function intercept_hook(ctx)
  -- Do not scan connections, initiated from the local
  -- host (divert == "output") by application under the name of
  -- "drweb" (group == "[drweb]")
  if ctx.divert == "output" and ctx.group == "drweb" then
    return "pass"
  end

  -- Do not scan connections, initiated from
  -- privileged ports (range is from 0 to 1024)
  if ctx.src.port >= 0 and ctx.src.port <= 1024 then
    return "pass"
  end

  -- Do not scan connections from local network IP addresses
  -- (IP address range 127.0.0.1/8)
  if ctx.src.ip.belongs("127.0.0.0/8") then
    return "pass"
  end

  -- Connection is scanned by default
  return "check"
end
```

Tables Used in Scripts

1. InterceptContext Table

The table is used to transfer data on the processed connection to the `intercept_hook` function. On the basis of this data one of the following action can be performed to the connection:

- skip without checking;
- disrupt the connection;
- send the connection to SpIDer Gate for checking.



Dr.Web Firewall for Linux fills the table with data. Some data in the table is already available by the time the `intercept_hook` functions is executed, other data (so called "lazy" data) will be calculated directly upon request of the corresponding field of the table.

Field	Description	Data type
<code>src</code>	Address and port of the client that initiated the connection Example: <pre>if ctx.src.port >= 0 and ctx.src.port <= 1024 then return "pass" end</pre>	TcpEndpoint table
<code>dst</code>	Address and port of the server, the connection to which was initiated by the client Example: <pre>if ctx.dst.ip.belongs("10.20.30.41/8") then return "reject" end</pre>	TcpEndpoint table
<code>divert</code>	The type of intercepted connection: <ul style="list-style-type: none">• "output"—outgoing connection;• "input"—incoming connection;• "forward"—transit connection. Example: <pre>if ctx.divert == "forward" then return "check" end</pre>	String
<code>iface_in</code>	The name of the interface from which the connection was initiated. If the name of the interface was not identified, it has the <code>nil</code> value.	String
<code>iface_out</code>	The name of the interface to which the packets were sent after the connection had been initiated. If the name of the interface was not identified, it has the <code>nil</code> value.	String
<code>uid</code>	The ID of the user who initiated the outgoing connection. If the connection type (<code>divert</code>) is not "output", or UID cannot be identified, it has <code>nil</code> value.	Number



Field	Description	Data type
<code>gid</code>	The ID of the group on behalf of which the outgoing connection was initiated. If the connection type (<code>divert</code>) is not "output", or GID cannot be identified, it has the <code>nil</code> value.	Number
<code>user</code>	The name of the user who initiated the outgoing connection. If the connection type (<code>divert</code>) is not "output", or UID cannot be identified, it has the <code>nil</code> value.	String
<code>group</code>	The name of the group on behalf of which the outgoing connection was initiated. If the connection type (<code>divert</code>) is not "output", or UID cannot be identified, it has the <code>nil</code> value.	String
<code>pid</code>	The ID of the process which initiated the outgoing connection. If the connection type (<code>divert</code>) is not "output", or PID cannot be identified, it has the <code>nil</code> value.	Number
<code>exe_path</code>	Executable path to the application file which initiated the outgoing connection. If the connection type (<code>divert</code>) is not "output", or executable path cannot be identified, it has the <code>nil</code> value.	String
Overridden metamethods: <i>None</i>		

2. TcpEndpoint Table

The table describes the address of connection point (client or server).

Field	Description	Data type
<code>ip</code>	IP address	The IpAddress table
<code>port</code>	Port number	Number
Overridden metamethods:		
<ul style="list-style-type: none">• <code>__tostring</code>—a function that converts <code>TcpEndpoint</code> to a string, for example: "127.0.0.1:443" (IPv4) or "[::1]:80" (IPv6);• <code>__concat</code>—a function that concatenates <code>TcpEndpoint</code> to a string		

Available Auxiliary Modules

For interconnection with Dr.Web for UNIX Internet Gateways in program space in Lua the following specific modules can be imported.



Name of the module	Function
drweb	The module that provides functions to record messages from the Lua program to the log of the Dr.Web for UNIX Internet Gateways component which has launched the Lua program and the means of asynchronous execution of Lua procedures
drweb.lookup	The module that provides tools to request data from external sources by calling the Dr.Web LookupD module

Contents of the drweb Module

1. Functions

The module provides a set of functions.

- Saving messages from the Lua program in the Dr.Web for UNIX Internet Gateways component log:
 - `log(<level>, <message>)` writes the `<message>` string to the Dr.Web for UNIX Internet Gateways log on the `<level>` level (the required level is defined using the `"debug"`, `"info"`, `"notice"`, `"warning"`, and `"error"`);
 - `debug(<message>)` writes the `<message>` string to the Dr.Web for UNIX Internet Gateways log at the `DEBUG` level;
 - `info(<message>)` writes the `<message>` string to the Dr.Web for UNIX Internet Gateways log at the `INFO` level;
 - `notice(<message>)` writes the `<message>` string to the Dr.Web for UNIX Internet Gateways log at the `NOTICE` level;
 - `warning(<message>)` writes the `<message>` string to the Dr.Web for UNIX Internet Gateways log at the `WARNING` level;
 - `error(<message>)` writes the `<message>` string to the Dr.Web for UNIX Internet Gateways log at the `ERROR` level.
- Managing the synchronization of Lua procedures:
 - `sleep(<sec.>)` pauses the execution of a Lua procedure instance for a specified number of seconds.
 - `async(<Lua function> [, <argument list>])` launches the specified function asynchronously and passes to it the specified argument list. The `async` function call completes immediately, and the return value (the table `Future`) allows you to obtain the result of the `<Lua function>`.
- Adding IP addresses to the [IpAddress](#) table:
 - `ip(<address>)` indicates an IP address, sent as the `<address>` string in the form of an `IpAddress` table. Either IPv4 or IPv6 addresses can be used.
- Uploading external data from a text file:
 - `load_set(<file path>)` generates a table with the `true` values from the contents of the



specified text file; strings read from a file are used as keys. Empty strings as well as strings with blank spaces will be ignored;

- `load_array(<file path>)` generates a string array from the contents of the specified text file. Empty strings and strings consisting of whitespace characters only, are ignored and are not included in the array.

2. Tables

- The `Future` table describes the pending result of performing a function using the `async` function.

Field	Description	Data type
<code>wait</code>	A function that returns the result of the function started using the <code>async.function</code> . If the function has not completed its execution yet, it waits for the completion and returns the result. If the function is completed before <code>wait</code> is called, the result is returned immediately. If the started function fails, the <code>wait</code> call generates the same error.	Function
Overridden metamethods: <i>None</i>		

- The `IpAddress` table describes an IP address.

Field	Description	Data type
<code>belongs</code>	<p>Function checks an IP address stored in the <code>IpAddress</code> table for belonging to the specified subnets (IP address ranges).</p> <p>Receives the only argument—a string that looks like: "<code><IP address></code>" or "<code><IP address>/<mask></code>", where <code><IP address></code>—a host address or a network address (for example, "<code>127.0.0.1</code>"), and <code><mask></code>—a subnetwork mask (can be specified as an IP address, for example, "<code>255.0.0.0</code>", or in the numerical form, for example, "<code>8</code>").</p> <p>Returns a Boolean value:</p> <ul style="list-style-type: none"> • <code>true</code> indicates that the address equals to at least one of the specified addresses or belongs at least one of the specified subnets (range of IP addresses); • <code>false</code>—otherwise. 	Function
Overridden metamethods:		
<ul style="list-style-type: none"> • <code>__toString</code> is a function that modifies <code>IpAddress</code> in a string, for example: "<code>127.0.0.1</code>" (IPv4) or "<code>:::1</code>" (IPv6); • <code>__concat</code> is a function that performs joining <code>IpAddress</code> to a string; • <code>__eq</code> is a function that checks the equality of two <code>IpAddress</code>; • <code>__band</code>—function that allows to apply a mask, for example: <code>dw.ip('192.168.1.2') & dw.ip('255.255.254.0')</code> 		



3. Examples

- Writing the messages generated by a procedure initiating asynchronously to the log:

```
local dw = require "drweb"

-- This function waits two seconds and returns a string,
-- received as an argument
function out_msg(message)
    dw.sleep(2)
    return message
end

-- "Main" function
function intercept(ctx)
    -- Output of a string at the NOTICE level to the Dr.Web for UNIX Internet
    Gateways log
    dw.notice("Intercept function started.")

    -- An asynchronous start of two copies of the out_msg function
    local f1 = dw.async(out_msg, "Hello,")
    local f2 = dw.async(out_msg, " world!")

    -- Waiting for the completion of the copies of the function
    -- out_msg and output its results to log
    -- the Dr.Web for UNIX Internet Gateways log at the DEBUG level
    dw.log("debug", f1.wait() .. f2.wait())
end
```

- Creating a scheduled procedure:

```
local dw = require "drweb"

-- Save the table Future in the future global variable in order
-- to preven the removal by the garbage collector
future = dw.async(function()
    while true do
        -- Everyday, the following message is displayed in the log
        dw.sleep(60 * 60 * 24)
        dw.notice("A brand new day began")
    end
end)
```

- Modifying an IP address represented as a string into an [IpAddress](#) table::

```
local dw = require "drweb"

local ipv4 = dw.ip("127.0.0.1")
local ipv6 = dw.ip(":::1")
local mapped = dw.ip("::ffff:127.0.0.1")
```



Contents of the drweb.lookup Module

1. Functions

The module provides the following functions:

- `lookup(<request>, <parameters>)` requests data from an external storage available via the Dr.Web LookupD module. The `<request>` argument must correspond to a section in the Dr.Web LookupD settings (the string `<type>@<tag>`). The `<parameters>` argument is optional. It describes substitutions that will be used to generate a request. The following automatically permitted markers can be used:
 - `$u`, `$U` is automatically replaced with `user`, the user name sent by the client component;
 - `$d`, `$D` is automatically replaced with `domain`, the domain sent by the client component.These arguments are set as a table. Keys and values of this table must be strings. The function returns an array of strings that are results of the request;
- `check(<checked string>, <request>, <parameters>)` returns `true` if `<checked string>` is found in the external repository, available via the Dr.Web LookupD module. The arguments `<request>` and `<parameters>` are equivalent to the arguments of the `lookup` function (see above). The `<checked string>` argument is supposed to be a string or a table with the `__tostring` metamethod (i.e. that can be formatted into a string).

2. Examples

- Writing to the log list of users retrieved from the `LookupD.LDAP.users` data source:

```
local dw = require "drweb"
local dwl = require "drweb.lookup"

-- "Main" function
function intercept(ctx)
  -- Writing the string at the NOTICE level to the Dr.Web for UNIX Internet
  Gateways log
  dw.notice("Intercept function started.")

  -- Writing the request results to the Dr.Web for UNIX Internet Gateways
  log
  -- to the 'ldap@users' data source
  for _, s in ipairs(dwl.lookup("ldap@users", {user="username"})) do
    dw.notice("Result for request to 'ldap@users': " .. s)
  end
end

end
```



Dr.Web ClamD

The Dr.Web ClamD component performs emulation using the 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 scans 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](#).

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 application requests to scan files or data streams. In the component settings, you can configure several connection points for external applications and adjust different scanning settings for each of the points, if required.

The 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: 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: 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 get documentation about this component of the product from the command line, use the following command: `man 1 drweb-clamd`.

Configuration Parameters

In this section

- [Component Parameters](#)
- [Special Aspects of Component Configuration](#)



The component uses configuration parameters which can be found in the [ClamD] section of the integrated [configuration file](#) of Dr.Web for UNIX Internet Gateways.

Component Parameters

The section contains the following parameters:

Parameter	Description
LogLevel <i>{logging level}</i>	<p>Logging level of the component.</p> <p>If the parameter value is not specified, the DefaultLogLevel parameter value from the [Root] section is used.</p> <p>Default value: Notice</p>
Log <i>{log type}</i>	<p>Logging method of the component.</p> <p>Default value: Auto</p>
ExePath <i>{path to file}</i>	<p>Executable path to the component.</p> <p>Default value: <opt_dir>/bin/drweb-clamd.</p> <ul style="list-style-type: none">• For GNU/Linux: /opt/drweb.com/bin/drweb-clamd.• For FreeBSD: /usr/local/libexec/drweb.com/bin/drweb-clamd
Start <i>{Boolean}</i>	<p>The component must be launched by the Dr.Web ConfigD configuration daemon.</p> <p>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.</p> <p>Default value: No</p>
Endpoint.<tag>.ClamdSocket <i>{IP address UNIX socket}</i>	<p>Create a new connection point naming it <tag> and allocates a socket (IPv4 address or address of a UNIX socket) for clients that need to scan files for threats.</p> <p>Only one socket can be specified for one <tag> point.</p> <p>Default value: not set</p>



Parameter	Description
<code>[Endpoint.<tag>.]DetectSuspicious</code> {Boolean}	<p>Inform about suspicious files detected by the heuristic analyzer.</p> <p>If the <code>Endpoint.<tag></code> prefix is specified, it means that the parameter value is set only for the <code><tag></code> connection point; otherwise, it is set for all points which do not have another value of this parameter specified for them.</p> <p>Default value: <code>Yes</code></p>
<code>[Endpoint.<tag>.]DetectAdware</code> {Boolean}	<p>Inform about files containing adware.</p> <p>If the <code>Endpoint.<tag></code> prefix is specified, it means that the parameter value is set only for the <code><tag></code> connection point; otherwise, it is set for all points which do not have another value of this parameter specified for them.</p> <p>Default value: <code>Yes</code></p>
<code>[Endpoint.<tag>.]DetectDialers</code> {Boolean}	<p>Inform about files containing dialers.</p> <p>If the <code>Endpoint.<tag></code> prefix is specified, it means that the parameter value is set only for the <code><tag></code> connection point; otherwise, it is set for all points which do not have another value of this parameter specified for them.</p> <p>Default value: <code>Yes</code></p>
<code>[Endpoint.<tag>.]DetectJokes</code> {Boolean}	<p>Inform about files containing jokes.</p> <p>If the <code>Endpoint.<tag></code> prefix is specified, it means that the parameter value is set only for the <code><tag></code> connection point; otherwise, it is set for all points which do not have another value of this parameter specified for them.</p> <p>Default value: <code>No</code></p>
<code>[Endpoint.<tag>.]DetectRiskware</code> {Boolean}	<p>Inform about files containing riskware.</p> <p>If the <code>Endpoint.<tag></code> prefix is specified, it means that the parameter value is set only for the <code><tag></code> connection point; otherwise, it is set for all points which do not have another value of this parameter specified for them.</p> <p>Default value: <code>No</code></p>
<code>[Endpoint.<tag>.]DetectHacktools</code> {Boolean}	<p>Inform about files containing hacktools.</p>



Parameter	Description
	<p>If the <code>Endpoint.<tag></code> prefix is specified, it means that the parameter value is set only for the <code><tag></code> connection point; otherwise, it is set for all points which do not have another value of this parameter specified for them.</p> <p>Default value: No</p>
<code>[Endpoint.<tag>.]ReadTimeout</code> <i>{time interval}</i>	<p>Maximum time-out to wait for data from a client.</p> <p>If the <code>Endpoint.<tag></code> prefix is specified, it means that the parameter value is set only for the <code><tag></code> connection point; otherwise, it is set for all points which do not have another value of this parameter specified for them.</p> <p>Default value: 5s</p>
<code>[Endpoint.<tag>.]StreamMaxLength</code> <i>{size}</i>	<p>Maximum size of data that can be received from a client (for transmitting data to scan as a stream of bytes).</p> <p>If the <code>Endpoint.<tag></code> prefix is specified, it means that the parameter value is set only for the <code><tag></code> connection point; otherwise, it is set for all points which do not have another value of this parameter specified for them.</p> <p>Default value: 25mb</p>
<code>[Endpoint.<tag>.]ScanTimeout</code> <i>{time interval}</i>	<p>Maximum time to scan one file (or one portion of data) received from a client.</p> <p>If the <code>Endpoint.<tag></code> prefix is specified, it means that the parameter value is set only for the <code><tag></code> connection point; otherwise, it is set for all points which do not have another value of this parameter specified for them.</p> <p>Acceptable values: from 1 second (1s) to 1 hour (1h).</p> <p>Default value: 3m</p>
<code>[Endpoint.<tag>.]HeuristicAnalysis</code> <i>{On Off}</i>	<p>Enable heuristic analysis for scanning.</p> <p>If the <code>Endpoint.<tag></code> prefix is specified, it means that the parameter value is set only for the <code><tag></code> connection point; otherwise, it is set for all points which do not have another value of this parameter specified for them.</p> <p>Default value: On</p>



Parameter	Description
<p>[Endpoint.<tag>.]PackerMaxLevel {integer}</p>	<p>Maximum nesting level for packed objects. A packed object is executable code compressed with special software (UPX, PELock, PECompact, Petite, ASPack, Morphine, etc.). Such objects may include other packed objects which may also include packed objects, etc. The maximum nesting level is the limit beyond which packed objects inside other packed objects are not scanned.</p> <p>If the Endpoint.<tag> prefix is specified, it means that the parameter 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.</p> <p>The value of this parameter can be any integer number greater than 0. If the value is set to 0, nested objects are not scanned.</p> <p>Default value: 8</p>
<p>[Endpoint.<tag>.]ArchiveMaxLevel {integer}</p>	<p>Maximum nesting level for archives (zip, rar, etc.) that can be scanned. Archives may include archives in which other archives may also be enclosed and so on. The maximum nesting level is the limit beyond which archives inside archives are not scanned.</p> <p>If the Endpoint.<tag> prefix is specified, it means that the parameter 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.</p> <p>The value of this parameter can be any integer number greater than 0. If the value is set to 0, nested objects are not scanned.</p> <p>Default value: 8</p>
<p>[Endpoint.<tag>.]MailMaxLevel {integer}</p>	<p>Maximum nesting level for files of mailers (pst, tbb and so on) in which other files may be enclosed (and these files may also include other files and so on). The value of this parameter specifies the nesting limit beyond which objects inside other objects will not be scanned.</p> <p>If the Endpoint.<tag> prefix is specified, it means that the parameter 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.</p>



Parameter	Description
	<p>The value of this parameter can be any integer number greater than 0. If the value is set to 0, nested objects are not scanned.</p> <p>Default value: 8</p>
<code>[Endpoint.<tag>.]ContainerMaxLevel</code> <i>{integer}</i>	<p>Maximum nesting level for other types of objects containing other objects (for instance, for HTML pages or jar files). The maximum nesting level is the limit beyond which objects inside objects are not scanned.</p> <p>If the <code>Endpoint.<tag></code> prefix is specified, it means that the parameter value is set only for the <code><tag></code> connection point; otherwise, it is set for all points which do not have another value of this parameter specified for them.</p> <p>The value of this parameter can be any integer number greater than 0. If the value is set to 0, nested objects are not scanned.</p> <p>Default value: 8</p>
<code>[Endpoint.<tag>.]</code> <code>MaxCompressionRatio</code> <i>{integer}</i>	<p>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.</p> <p>The compression ratio must not be smaller than 2.</p> <p>Default value: 500</p>

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 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 from 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 emulation of the `clamd` interface makes it possible to integrate Dr.Web ClamD with external applications capable to connect to the `clamd` anti-virus daemon (included in ClamAV).



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 <code>squidclamav</code> or HAVP as an intermediate component. Links to documentation Squid documentation: http://www.squid-cache.org/Doc/ . Description and source code files of <code>squidclamav</code> : https://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 <code>clamd</code> for anti-virus scanning configuration: <pre>ENABLECLAMD true CLAMDSOCKET <path_to_clamd_UNIX_socket></pre> or (if TCP connection is used instead of a UNIX socket): <pre>ENABLECLAMD true CLAMDSERVER <IP> CLAMDPORTR <port></pre> where <code><path_to_clamd_UNIX_socket></code> or the <code><IP>:<port></code> pair corresponds to the socket of a connection point (<i>endpoint</i>) configured in Dr.Web ClamD. Links to documentation HAVP documentation: http://www.havp.org/documentation/

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]
ClamSocket = /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 scanning component Dr.Web File Checker is designed for scanning files and directories in the file system. It is used by other components of Dr.Web for UNIX Internet Gateways to scan file system objects. Moreover, this component also functions as a quarantine manager, as it manages the contents of the directories where isolated 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 scanned files and directories and saves all the data about the objects that have been checked to a special cache to avoid repeated scanning of objects that have been already scanned and have not been modified since that (in this case, if a request to scan such an object is received, the previous scan result, retrieved from cache, is returned).

When requests to check file system objects are received from Dr.Web for UNIX Internet Gateways 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 puts it into detected threats registry and neutralizes it (cures, deletes or quarantines) if this action has been specified by the client component that initiated the scanning as the reaction to a threat. The 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 scan 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” scanning method using the `flowscan` command of the [Dr.Web Ctl](#) utility (launched with the `drweb-ctl` command). However, for a normal on-demand scanning, it is recommended that you use the `scan` command.

During its work, the file scanning component not only keeps a threats registry and manages quarantine, but also collects overall file scan statistics, averaging the number of files checked within a second in the last minute, last 5 minutes, last 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
<code>--help</code>	Function: Output short help information about command-line parameters to the console or to the terminal emulator and to exit upon completion. Short form: <code>-h</code> Arguments: None.
<code>--version</code>	Function: Output information about the version of this component to the console or to the terminal emulator and to exit after completion. Short form: <code>-v</code> 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 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 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 get 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 can be in the [FileCheck] section of the integrated [configuration file](#) of Dr.Web for UNIX Internet Gateways.

This section stores the following parameters:

Parameter	Description
LogLevel <i>{logging level}</i>	<p>Logging level of the component.</p> <p>If the parameter value is not specified, the DefaultLogLevel parameter value from the [Root] section is used.</p> <p>Default value: Notice</p>
Log <i>{log type}</i>	<p>Logging method of the component.</p> <p>Default value: Auto</p>
ExePath <i>{path to file}</i>	<p>Executable path to the component.</p> <p>Default value: <opt_dir>/bin/drweb-filecheck.</p> <ul style="list-style-type: none">• For GNU/Linux: /opt/drweb.com/bin/drweb-filecheck.• For FreeBSD: /usr/local/libexec/drweb.com/bin/drweb-filecheck
DebugClientIpc <i>{Boolean}</i>	<p>Indicates whether detailed IPC messages should be included into the log file on the debug level (i.e. when LogLevel = DEBUG).</p> <p>Default value: No</p>
DebugScan <i>{Boolean}</i>	<p>Write detailed messages received during file scanning to the log file on the debug level (i.e. when LogLevel = DEBUG).</p> <p>Default value: No</p>
DebugFlowScan <i>{Boolean}</i>	<p>Write detailed messages about file scanning by the "flow" method to the log file on the debug level (i.e. when LogLevel = DEBUG).</p> <p>Default value: No</p>
DebugProxyScan <i>{Boolean}</i>	<p>Write detailed messages about file scanning by the "proxy" method to the log file on the debug level (i.e. when LogLevel = DEBUG). Normally this scanning method is used by the Dr.Web ClamD component.</p> <p>Default value: No</p>
DebugCache <i>{Boolean}</i>	<p>Write detailed messages about the cached results of scanning should be included to the log file on the debug level (i.e. when LogLevel = DEBUG).</p> <p>Default value: No</p>
MaxCacheSize <i>{size}</i>	<p>Maximum allowed size of cache to store data about scanned files.</p> <p>If 0 is specified, caching is disabled.</p>



Parameter	Description
	Default value: 50mb
RescanInterval <i>{time interval}</i>	<p>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).</p> <p>Acceptable values: from 0 seconds (0s) to 1 minute (1m) inclusive. If the set interval is less than 1s—there will be no delay, the file will be scanned upon any request.</p> <p>Default value: 1s</p>
IdleTimeLimit <i>{time interval}</i>	<p>Maximum idle time for the component. When the specified period of time expires, the component shuts down.</p> <p>Acceptable values: from 10 seconds (10s) to 30 days (30d) inclusive. If the None value is set, the component will functionate eternally; the SIGTERM signal will not be sent if the components goes idle.</p> <p>Default value: 30s</p>



Dr.Web Network Checker

Network checker agent Dr.Web Network Checker is designed for scanning the data received through the network in the scan 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 ClamD, SpiDer Gate, Dr.Web ICAPD).



This component is not designed to organize distributed scanning of files located in the local file system, since it cannot replace the Dr.Web File Checker component. To organize distributed scanning of local files, use [Dr.Web MeshD](#) component.

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. 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 component allows sending the data not represented as files in the local file system for scanning into [Dr.Web Scanning Engine](#) engine (located on local or remote host). This data is processed by the components which send data for scanning via (Dr.Web ICAPD, Dr.Web ClamD) connection. Mind that these components always use Dr.Web Network Checker for files transmission to Dr.Web Scanning Engine engine, even if it is located on local host. Thus, if Dr.Web Network Checker is unavailable, *these components cannot work correctly*.

In addition, Dr.Web Network Checker allows the Dr.Web for UNIX Internet Gateways connection with a given set of nodes on the network with the Dr.Web for UNIX Internet Gateways installed on them (or any other Dr.Web for UNIX solution 10.1 or later) in order to organize distributed



checks data presence which is not represented as files in the local file system. Thereby, this component allows to create and configure a *scanning cluster*, which is a set of network nodes that exchange data for verification (each instance must have its own instance of the Dr.Web Network Checker distributed verification agent). On each node of the network included in the scanning cluster, Dr.Web Network Checker performs automatic distribution of tasks for scanning data, transferring it over the network to all available nodes with which the connection is configured. At the same time, the load balancing on nodes is performed, caused by data verification depending on the amount of resources available on remote nodes (as an indicator of the amount of resources available for load, the number of child scanning processes generated by the scanning core Dr.Web Scanning Engine on this node is used). The lengths of the files queues waiting for checking on each used node are also estimated.

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.

The data received via network for scanning is saved to the local file system as temporary files and are sent to [Dr.Web Scanning Engine](#) engine or, in case if it is unavailable, to the other node of the scanning cluster.

The `InternalOnly` parameter, which you can find in [settings](#), allows to manage the Dr.Web Network Checker operation mode. It indicates if Dr.Web Network Checker is used for including Dr.Web for UNIX Internet Gateways to the scanning cluster or for internal purposes of the Dr.Web for UNIX Internet Gateways local components only.



You can 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.

The example of creating the scanning cluster can be found in [Creating the Scanning Cluster](#) section.

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
-----------	-------------



<code>--help</code>	Function: Output short help information about command-line parameters to the console or to the terminal emulator and to exit upon completion. Short form: <code>-h</code> Arguments: None.
<code>--version</code>	Function: Output information about the version of this component to the console or to the terminal emulator and to exit after completion. Short form: <code>-v</code> 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, in the component [configuration](#), a value of the `FixedSocket` parameter is specified and the `InternalOnly` parameter is set to `No`, 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 with the `drweb-ctl` [command](#)). 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 get 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 can be found in the `[NetCheck]` section of the integrated [configuration file](#) of Dr.Web for UNIX Internet Gateways.



The section contains the following parameters:

Parameter	Description
<code>LogLevel</code> <i>{logging level}</i>	<p>Logging level of the component.</p> <p>If the parameter value is not specified, the <code>DefaultLogLevel</code> parameter value from the <code>[Root]</code> section is used.</p> <p>Default value: <code>Notice</code></p>
<code>Log</code> <i>{log type}</i>	<p>Logging method of the component.</p> <p>Default value: <code>Auto</code></p>
<code>ExePath</code> <i>{path to file}</i>	<p>Executable path to the component.</p> <p>Default value: <code><opt_dir>/bin/drweb-netcheck</code>.</p> <ul style="list-style-type: none">• For GNU/Linux: <code>/opt/drweb.com/bin/drweb-netcheck</code>.• For FreeBSD: <code>/usr/local/libexec/drweb.com/bin/drweb-netcheck</code>
<code>FixedSocket</code> <i>{path to file address}</i>	<p>Socket of the fixed Dr.Web Network Checker agent instance.</p> <p>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.</p> <p>Allowed values:</p> <ul style="list-style-type: none">• <code><path to file></code> is a path to a local UNIX socket;• <code><address></code> is a network socket as a pair <code><IP address>:<port></code>. <p>Default value: <i>(not set)</i></p>
<code>InternalOnly</code> <i>{Boolean}</i>	<p>Managing the operation mode of the component.</p> <p>If the value is set to <code>Yes</code>, the component is used for internal purposes of the Dr.Web for UNIX Internet Gateways components only and it is not used for including Dr.Web for UNIX Internet Gateways to the scanning cluster and for processing external (to the Dr.Web for UNIX Internet Gateways) client applications, regardless of the <code>LoadBalance*</code> settings and the value of the <code>FixedSocket</code> parameter.</p> <p>Default value: <code>No</code></p>
<code>RunAsUser</code> <i>{UID user name}</i>	<p>The name of the user on whose behalf the component is 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: <code>RunAsUser = name:123456</code>.</p>



Parameter	Description
	<p>When a user name is not specified, the component operation terminates with an error after the startup.</p> <p>Default value: <code>drweb</code></p>
<code>IdleTimeLimit</code> <i>{time interval}</i>	<p>Maximum idle time for the component. If the specified value is exceeded, the component shuts down.</p> <p>If the <code>LoadBalanceAllowFrom</code> or <code>FixedSocket</code> parameter is set, this setting is ignored (the component does not finish its operation after the time interval expires).</p> <p>Acceptable values: from 10 seconds (10s) to 30 days (30d) inclusive.</p> <p>If the <code>None</code> value is set, the component will functionate eternally; the <code>SIGTERM</code> signal will not be sent if the components goes idle.</p> <p>Default value: <code>30s</code></p>
<code>LoadBalanceUseSsl</code> <i>{Boolean}</i>	<p>Use SSL/TLC for connecting to other hosts.</p> <p>Allowed values:</p> <ul style="list-style-type: none">• <code>Yes</code>—use SSL/TLS;• <code>No</code>—do not use SSL/TLS. <p>If the parameter is set to <code>Yes</code>, a certificate and the corresponding private key should be specified for this host and for hosts with which it interacts (the parameters <code>LoadBalanceSslCertificate</code> and <code>LoadBalanceSslKey</code>).</p> <p>Default value: <code>No</code></p>
<code>LoadBalanceSslCertificate</code> <i>{path to file}</i>	<p>Path to the SSL certificate used by Dr.Web Network Checker for communication with other hosts via a secure SSL/TLS connection.</p> <p>Please note that the certificate file and the private key file (which is specified by a parameter described below) must form a matching pair.</p> <p>Default value: <i>(not set)</i></p>
<code>LoadBalanceSslKey</code> <i>{path to file}</i>	<p>Path to the private key used by Dr.Web Network Checker for communication with other hosts via a secure SSL/TLS connection.</p> <p>Please note that the certificate file and the private key file (which is specified by the mentioned parameter) must form a matching pair.</p> <p>Default value: <i>(not set)</i></p>
<code>LoadBalanceSslCa</code> <i>{path}</i>	<p>The path to the directory or file with the list of trusted root certificates. Among these certificates, there must be a certificate that certifies the authenticity of the certificates used by agents</p>



Parameter	Description
	<p>within the scanning cluster when exchanging data over SSL/TLS protocols.</p> <p>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.</p> <p>Default value: <i>(not set)</i></p>
<code>LoadBalanceSslCrl</code> <i>{path}</i>	<p>Path to the directory or file with system list of revoked certificates.</p> <p>If the parameter value is not specified, Dr.Web Network Checker running on this host does not check the certificates of the interacting agents for validity, but they may check the validity of the certificate used by the agent running on this host, depending on the settings.</p> <p>Default value: <i>(not set)</i></p>
<code>LoadBalanceServerSocket</code> <i>{address}</i>	<p>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).</p> <p>Default value: <i>(not set)</i></p>
<code>LoadBalanceAllowFrom</code> <i>{IP address}</i>	<p>IP address of a remote network host from which the Dr.Web Network Checker receives files for scanning (as a scanning server).s</p> <p>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).</p> <p>Example: Add to the list of host addresses 192.168.0.1 and 10.20.30.45.</p> <ol style="list-style-type: none">Adding of values to the configuration file.<ul style="list-style-type: none">Two values in a line:<pre>[NetCheck] LoadBalanceAllowFrom = "192.168.0.1", "10.20.30.45"</pre>Two lines (a value per line):<pre>[NetCheck] LoadBalanceAllowFrom = 192.168.0.1 LoadBalanceAllowFrom = 10.20.30.45</pre>



Parameter	Description
	<p>2. Adding values via the <code>drweb-ctl cfset</code> command:</p> <pre data-bbox="703 297 1449 465"># drweb-ctl cfset NetCheck.LoadBalanceAllowFrom -a 192.168.0.1 # drweb-ctl cfset NetCheck.LoadBalanceAllowFrom -a 10.20.30.45</pre> <p>If the parameter is empty, removed files cannot be received for scanning (the host does not operate as a scanning server).</p> <p>Default value: <i>(not set)</i></p>
LoadBalanceSourceAddress <i>{IP address}</i>	<p>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).</p> <p>If an empty value is specified, the network interface automatically selected by the OS kernel is used.</p> <p>Default value: <i>(not set)</i></p>
LoadBalanceTo <i>{address}</i>	<p>Socket (IP address or port) of a remote host to which Dr.Web Network Checker on the host can send files for their remote scanning (as a network scanning client).</p> <p>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).</p> <p>Example: Add sockets 192.168.0.1:1234 and 10.20.30.45:5678 to the list.</p> <p>1. Adding of values to the configuration file.</p> <ul style="list-style-type: none">• Two values in one string: <pre data-bbox="703 1435 1449 1570">[NetCheck] LoadBalanceTo = "192.168.0.1:1234", "10.20.30.45:5678"</pre> <ul style="list-style-type: none">• Two strings (one value per a string): <pre data-bbox="703 1630 1449 1765">[NetCheck] LoadBalanceTo = 192.168.0.1:1234 LoadBalanceTo = 10.20.30.45:5678</pre> <p>2. Adding values with the <code>drweb-ctl cfset</code> command:</p> <pre data-bbox="703 1832 1449 2000"># drweb-ctl cfset NetCheck.LoadBalanceTo -a 192.168.0.1:1234 # drweb-ctl cfset NetCheck.LoadBalanceTo -a 10.20.30.45:5678</pre>



Parameter	Description
	<p>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).</p> <p>Default value: <i>(not set)</i></p>
<code>LoadBalanceStatusInterval</code> <i>{time interval}</i>	<p>Time interval considered by the host to send the next message containing information about its workload to all scanning clients (specified in the <code>LoadBalanceAllowFrom</code> parameter).</p> <p>Default value: 1s</p>
<code>SpoolDir</code> <i>{path to directory}</i>	<p>Local file system directory used to store files sent over the network for scanning and received by Dr.Web Network Checker.</p> <p>Default value: <code>/tmp/netcheck</code></p>
<code>LocalScanPreference</code> <i>{fractional number}</i>	<p>Relative weight (priority) of the 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.</p> <p>Minimum value: 1.</p> <p>Default value: 1</p>

Creating the Scanning Cluster

In this section:

- [Introductory Remarks.](#)
- [The Example of Creating the Scanning Cluster.](#)
- [Configuring Cluster Nodes.](#)
- [Verifying the Cluster Operability.](#)

Introductory Remarks

To create the scanning cluster that allows to perform the distributed checks (while scanning files or other objects), you need to have a set of network nodes with the installed Dr.Web Network Checker component on each node. To make the cluster node not only to receive and transmit data to be scanned, it is also necessary to have the scan engine Dr.Web Scanning Engine installed on the node. Thus, to create the node of the scanning cluster, it is necessary that the minimum set of the following components is installed (minimally) on the server (other components of Dr.Web for UNIX Internet Gateways which are installed automatically to ensure the functionality of the components listed here, are skipped):

1. Dr.Web Network Checker (`drweb-netcheck` package) is a component that provides networking between nodes;

2. [Dr.Web Scanning Engine](#) (`drweb-se` package) is the scan engine that is need for scanning data received via network. The component may be absent, in this is node will only transmit data to be checked to other scanning cluster nodes.

The nodes that constitute the scanning cluster form *peer to peer* network, i.e. each of the nodes, depending on which [settings](#) are defined in the Dr.Web Network Checker component on this node, is able to act as either a *scanning client* (which transmits data for scanning to other nodes) or as a *scanning server* (which receives data for scanning from other nodes). With the appropriate settings, the cluster node can be both the scanning client and the scanning server at the same time.

The Dr.Web Network Checker parameters, related to scanning cluster setting, have names starting with `LoadBalance`.

The Example of Creating the Scanning Cluster

Study the example of creating the scanning cluster, displayed on the figure below.

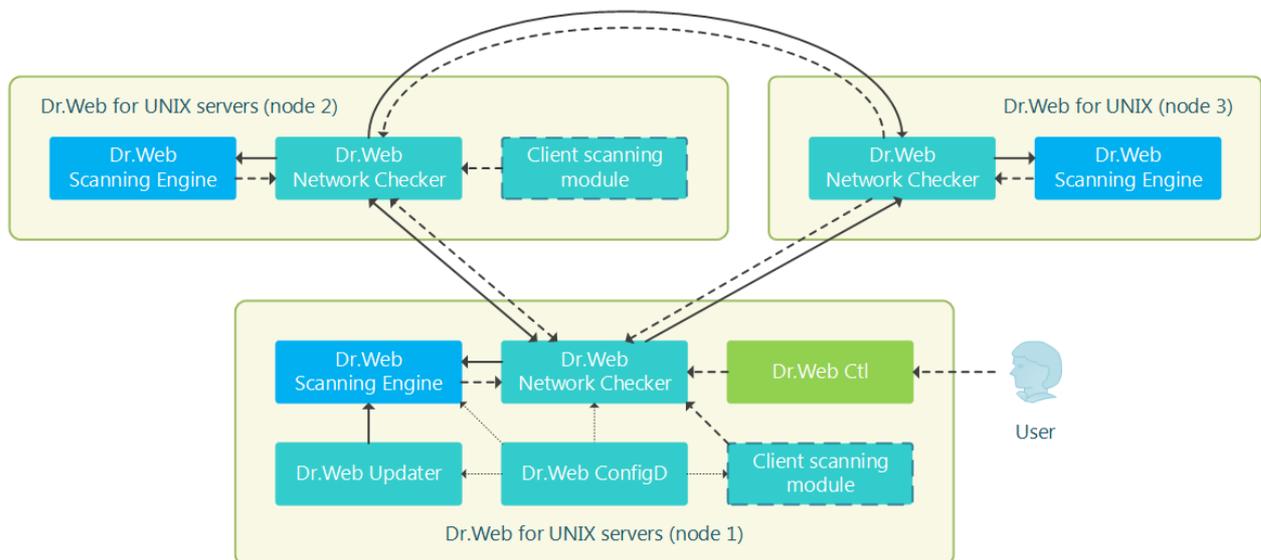


Figure 13. The scanning cluster structure

In this case, it is assumed that the cluster consists of three nodes (displayed on the figure as *node 1*, *node 2*, and *node 3*). In this case, node 1 and node 2 are servers with a full-fledged Dr.Web product for UNIX servers installed (for example, Dr.Web for UNIX file servers or Dr.Web for UNIX internet gateways, the product type does not matter), and node 3 is used only for assistance in scanning files transferred from nodes 1 and 2. Therefore, only the minimum required component set (Dr.Web Network Checker and Dr.Web Scanning Engine) is installed, other components that are automatically installed to ensure the node operability, such as Dr.Web ConfigD, are not displayed on the figure). Nodes 1 and 2 can work both as servers and scanning clients between each other (perform mutual distribution of the load, associated with scanning), and node 3—only as a server, receiving tasks from nodes 1 and 2.



These components will be distributed between the locally installed scan engine Dr.Web Scanning Engine and the cluster partner nodes, acting as scanning servers depending on the load balance.



It is important to note that only components scan data that is not represented as files in the local file system, can act as a client module of verification. This means that the scanning cluster cannot be used for distributed scanning of files by SplDer Guard file system monitors and by the Dr.Web File Checker component.

Configuring Cluster Nodes

To customize the specified cluster configuration you need to change Dr.Web Network Checker settings on all cluster nodes. All following settings are given as `.ini` file (refer to configuration file [format](#) description).

Node 1

```
[NetCheck]
InternalOnly=No
LoadBalanceUseSsl = No
LoadBalanceServerSocket = <Node 1 IP address> : <Node 1 port>
LoadBalanceAllowFrom = <Node 2 IP address>
LoadBalanceSourceAddress = <Node 1 IP address>
LoadBalanceTo = <Node 2 IP address> : <Node 2 port>
LoadBalanceTo = <Node 3 IP address> : <Node 3 port>
```

Node 2

```
[NetCheck]
InternalOnly=No
LoadBalanceUseSsl = No
LoadBalanceServerSocket = <Node 2 IP address> : <Node 2 port>
LoadBalanceAllowFrom = <Node 1 IP address>
LoadBalanceSourceAddress = <Node 2 IP address>
LoadBalanceTo = <Node 1 IP address> : <Node 1 port>
LoadBalanceTo = <Node 3 IP address> : <Node 3 port>
```

Node 3

```
[NetCheck]
InternalOnly=No
LoadBalanceUseSsl = No
LoadBalanceServerSocket = <Node 3 IP address> : <Node 3 port>
LoadBalanceAllowFrom = <Node 1 IP address>
LoadBalanceAllowFrom = <Node 2 IP address>
```



Notes:

- Other (not mentioned here) Dr.Web Network Checker parameters are left unchanged.
- IP addresses and port numbers should be changed to real.
- Using of SSL for data exchange between nodes in this example is disabled. If you need to use SSL, you must set the value `Yes` for `LoadBalanceUseSsl` parameter, as well as set the needed values for the following parameters `LoadBalanceSslCertificate`, `LoadBalanceSslKey` and `LoadBalanceSslCa`.

Verifying the Cluster Operability

To check the cluster operation in data distribution mode, use the following [command](#) on nodes 1 and 2:

```
$ drweb-ctl netscan <path to file or directory>
```

When executing the specified command, files from the specified directory should be checked by Dr.Web Network Checker, which should distribute the check to customized cluster nodes. To view statistics of network checks on each node before scanning, run the display of the statistics of Dr.Web Network Checker using the following [command](#) (to interrupt the displaying of statistics press CTRL+C):

```
$ drweb-ctl stat -n
```



Dr.Web Scanning Engine

The Dr.Web Scanning Engine scan 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 scan 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 scan engine operates in the daemon mode, as a service which receives scanning requests from other Dr.Web for UNIX Internet Gateways components (these are Dr.Web File Checker and Dr.Web Network Checker, and, partially, Dr.Web MeshD). If Dr.Web Scanning Engine and Dr.Web Virus-Finding Engine are absent or unavailable, no anti-virus scanning is performed on this node (except for Dr.Web for UNIX Internet Gateways contains the Dr.Web MeshD component, which settings contain the connection to local cloud nodes, providing scan engine services).

Operating Principles

The component operates as a service which receives requests to scan file system objects (files and boot disk records) from the Dr.Web for UNIX Internet Gateways components on embedded threats. It also queues scanning tasks and scans requested objects by using the Dr.Web Virus-Finding Engine scan engine. If a threat is detected and it must be cured according to the scanning task, the scan engine attempts to cure it if this action can be applied to the scanned object.

The scanning engine, the Dr.Web Virus-Finding Engine scan engine, and the virus databases form one unit and cannot be separated: the scan engine downloads virus databases and provides the operation environment for the cross-platform scan engine Dr.Web Virus-Finding Engine. The virus databases and the scan engine are updated by the [Dr.Web Updater](#) update component that is included in Dr.Web for UNIX Internet Gateways, but this component is not a part of the scan 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 the centralized protection mode, updating of virus databases and the scan engine is performed by the [Dr.Web ES Agent](#). The latter component interacts with the centralized protection server and receives the updates.

The Dr.Web 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 anti-virus databases are up to date. In the latter case, the engine startup and the updating of virus databases is performed by an external application that uses the engine. The Dr.Web for UNIX Internet Gateways components that issue requests to the scan engine asking it to scan files use the same interface as other external applications.



Users are provided with the opportunity to create own component (external application) using Dr.Web Scanning Engine 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 scan 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 scan engine uses the average length of queues per second. This rate is averaged for the last minute, last 5 minutes and last 15 minutes.

The Dr.Web Virus-Finding Engine scan 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.

The Dr.Web Virus-Finding Engine scan engine allows to scan and cure both files and packed objects or objects in different containers (such as archives, email messages, and so on).

Command-Line Arguments

To run the scan 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
<code>--help</code>	Function: Output short help information about command-line parameters to the console or to the terminal emulator and to exit upon completion. Short form: <code>-h</code> Arguments: None.
<code>--version</code>	Function: Output information about the version of this component to the console or to the terminal emulator and to exit after completion. Short form: <code>-v</code> Arguments: None.
<i>Additional launch parameters (they are the same as configuration file parameters and substitute them when required):</i>	
<code>--CoreEnginePath</code>	Function: Specify the path to the library of the Dr.Web Virus-Finding Engine scan engine. Short form: No. Arguments: <i><path to the file></i> —a full path to the library that you want to use.
<code>--VirusBaseDir</code>	Function: Specify the path to the directory with virus database files. Short form: No. Arguments: <i><path to the directory></i> —path to the virus database directory.
<code>--TempDir</code>	Function: Specify the path to the directory with temporary files. Short form: No. Arguments: <i><path to the directory></i> —full path to the directory with temporary files.
<code>--Key</code>	Function: Specify the path to the key file. Short form: No. Arguments: <i><path to the file></i> —a full path to the key file that you want to use.
<code>--MaxForks</code>	Function: Specify the maximum allowed number of child processes which can be started by Dr.Web Scanning Engine during scanning. Short form: No. Arguments: <i><number></i> —the maximum allowed number of child processes.
<code>--WatchdogInterval</code>	Description: Frequency with which Dr.Web Scanning Engine checks whether child processes are operable and stops those processes that stopped responding. Short form: No. Arguments: <i><time interval></i> —frequency of checking child processes.
<code>--Shelltrace</code>	Function: Turn on the shell tracing (log detailed information on file scanning performed by Dr.Web Virus-Finding Engine). Short form: No. Arguments: None.



<code>--LogLevel</code>	<p>Description: Set the level of logging executed by Dr.Web Scanning Engine during the operation.</p> <p>Short form: No.</p> <p>Arguments: <i><logging level></i>. Allowed values:</p> <ul style="list-style-type: none">• <code>DEBUG</code>—the most detailed logging level. All messages and debug information are registered.• <code>INFO</code>—all messages are registered.• <code>NOTICE</code>—all error messages, warnings, and notifications are registered.• <code>WARNING</code>—all error messages and warnings are registered.• <code>ERROR</code>—only error messages are registered.
<code>--Log</code>	<p>Description: Specify the method for logging component messages.</p> <p>Short form: No.</p> <p>Arguments: <i><log type></i>. Allowed values:</p> <ul style="list-style-type: none">• <code>Stderr[:ShowTimestamp]</code>—messages are output to a standard error stream <i>stderr</i>. The additional option <code>ShowTimestamp</code> is used to add a time stamp to every message.• <code>Syslog[:<facility>]</code>—messages are transmitted to the system logging service <code>syslog</code>. Additional option <i><facility></i> is used to specify a level at which <code>syslog</code> registers messages. The following values are possible:<ul style="list-style-type: none">○ <code>DAEMON</code>—messages of daemons.○ <code>USER</code>—messages of user processes.○ <code>MAIL</code>—messages of mail programs.○ <code>LOCAL0</code>—messages of local processes 0....○ <code>LOCAL7</code>—messages of local processes 7.• <i><path></i>—path to the file where all messages are registered. <p>Examples:</p> <pre>--Log /var/opt/drweb.com/log/se.log --Log Stderr:ShowTimestamp --Log Syslog:DAEMON</pre>

Example:

```
$ /opt/drweb.com/bin/drweb-se /tmp/drweb.ipc/.se --MaxForks=5
```

This command starts an instance of the Dr.Web Scanning Engine scan engine, creates the `/tmp/drweb.ipc/.se` UNIX socket for the interaction with the client components and limits the number of child scanning processes to 5.



Startup Notes

When necessary, any number of instances of the Dr.Web Scanning Engine scan engine can be started. The instances provide the scanning service for client applications (not only for the Dr.Web for UNIX Internet Gateways components). At that, if a value of the `FixedSocketPath` parameter is specified in the component [configuration](#), one instance of the scan 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 scan 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 get 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 can be found in the `[ScanEngine]` section of the integrated [configuration file](#) of Dr.Web for UNIX Internet Gateways.

This section stores the following parameters:

Parameter	Description
<code>LogLevel</code> <i>{logging level}</i>	Logging level of the component. If the parameter value is not specified, the <code>DefaultLogLevel</code> parameter value from the <code>[Root]</code> section is used. Default value: <code>Notice</code>
<code>Log</code> <i>{log type}</i>	Logging method of the component. Default value: <code>Auto</code>
<code>ExePath</code> <i>{path to file}</i>	Executable path to the component. Default value: <code><opt_dir>/bin/drweb-se</code> . <ul style="list-style-type: none">• For GNU/Linux: <code>/opt/drweb.com/bin/drweb-se</code>.• For FreeBSD: <code>/usr/local/libexec/drweb.com/bin/drweb-se</code>



Parameter	Description
<code>FixedSocketPath</code> <i>{path to file}</i>	<p>Path to the UNIX socket of the fixed instance of the Dr.Web Scanning Engine scan engine.</p> <p>If this parameter is specified, the Dr.Web ConfigD configuration daemon checks that there is always a running component copy of the scan engine that is available to the clients via this socket.</p> <p>Default value: <i>(not set)</i></p>
<code>IdleTimeLimit</code> <i>{time interval}</i>	<p>Maximum idle time for the component. When the specified period of time expires, the component shuts down.</p> <p>If the <code>FixedSocketPath</code> parameter is set, this setting is ignored (the component does not finish its operation after the time interval expires).</p> <p>Acceptable values: from 10 seconds (10s) to 30 days (30d) inclusive. If the <code>None</code> value is set, the component will functionate eternally; the <code>SIGTERM</code> signal will not be sent if the components goes idle.</p> <p>Default value: 1h</p>
<code>MaxForks</code> <i>{integer}</i>	<p>Maximum allowed number of child processes run by Dr.Web Scanning Engine, which can be run simultaneously.</p> <p>Default value: <i>Automatically determined</i> as twice the number of available CPU cores; or 4, if the resulting number is less than 4.</p>
<code>BufferedIo</code> <i>{On Off}</i>	<p>Use buffered input/output (I/O) when scanning files.</p> <p>Using buffered I/O in the FreeBSD and GNU/Linux OSes can increase scanning speed of the files on slow disks.</p> <p>Default value: <code>Off</code></p>
<code>WatchdogInterval</code> <i>{time interval}</i>	<p>Rate at which Dr.Web Scanning Engine checks whether child processes are operable in order to detect processes that stopped responding ("watchdog").</p> <p>Default value: 1.5s</p>



Dr.Web Updater

The update component Dr.Web Updater is designed for receiving all available updates for virus databases and the Dr.Web Virus-Finding Engine scan engine from Doctor Web update servers and synchronize updates with local cloud of Dr.Web for UNIX products (via the [Dr.Web MeshD](#), if it is present in the product) component.

If Dr.Web for UNIX Internet Gateways operates in the [centralized protection mode](#), the updates are received from the centralized 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 (updates synchronization with a local cloud of Dr.Web for UNIX products is not produced either).

Operating Principles

The component is designed to establish connections to Doctor Web update servers to check for updates for virus databases and the Dr.Web Virus-Finding Engine scan 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). Only basic and digest authentication are supported for connection to Doctor Web update servers.

If Dr.Web for UNIX Internet Gateways is not connected to the centralized protection server or is connected to the server in the mobile mode, Dr.Web Updater is automatically started by the Dr.Web ConfigD configuration daemon. The 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).

When updates become available on the servers, they are downloaded to the `<var_dir>/cache` directory (for GNU/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 `*Dr1Dir` parameters, grouped by the update type: for virus databases and the scan engine, database of web resource categories (these files are grouped according to the component which is updated—virus databases and the scan engine, the anti-spam component). Upon 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 `*CustomDr1Dir` 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 `*CustomDr1Dir` 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 [Dr.Web Ctl](#) command line (it is run by `drweb-ctl` command).

If Dr.Web for UNIX Internet Gateways is connected to the local cloud of Dr.Web for UNIX products, and it is not connected to the centralized protection server, the Dr.Web Updater component is used to synchronize updates received by cloud hosts as well, that is, it transmits updates received from update servers to the cloud, and receives updates from the cloud, which allows to reduce the total load on the Dr.Web update server. This option can be enabled or disabled in the component [settings](#).

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
<code>--help</code>	Function: Output short help information about command-line parameters to the console or to the terminal emulator and to exit upon completion. Short form: <code>-h</code> Arguments: None.
<code>--version</code>	Function: Output information about the version of this component to the console or to the terminal emulator and to exit after completion. Short form: <code>-v</code> 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 the scan 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 get 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 can be found in the [Update] section of the integrated [configuration file](#) of Dr.Web for UNIX Internet Gateways.

The section contains the following parameters:

Parameter	Description
<code>LogLevel</code> <i>{logging level}</i>	Logging level of the component. If the parameter value is not specified, the <code>DefaultLogLevel</code> parameter value from the [Root] section is used. Default value: <code>Notice</code>
<code>Log</code> <i>{log type}</i>	Logging method of the component. Default value: <code>Auto</code>
<code>ExePath</code> <i>{path to file}</i>	Executable path to the component. Default value: <code><opt_dir>/bin/drweb-update</code> . <ul style="list-style-type: none">• For GNU/Linux: <code>/opt/drweb.com/bin/drweb-update</code>.• For FreeBSD: <code>/usr/local/libexec/drweb.com/bin/drweb-update</code>
<code>RunAsUser</code> <i>{UID user name}</i>	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: <code>RunAsUser = name:123456</code> . When a user name is not specified, the component operation terminates with an error after the startup. Default value: <code>drweb</code>



Parameter	Description
<code>UpdateInterval</code> <i>{time interval}</i>	<p>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.</p> <p>Default value: 30m</p>
<code>RetryInterval</code> <i>{time interval}</i>	<p>Frequency of repeated attempts to perform an update using the update servers if the previous attempt failed.</p> <p>Acceptable values: from 1 minute (1m) to 30 minutes (30m) inclusive.</p> <p>Default value: 3m</p>
<code>MaxRetries</code> <i>{integer}</i>	<p>Number of repeated attempts to perform an update using the update servers (at the rate specified in <code>RetryInterval</code>) if the previous attempt failed.</p> <p>If the value is set to 0, repeated attempts are not made (the next update will be performed after the time period specified in <code>UpdateInterval</code>).</p> <p>Default value: 3</p>
<code>Proxy</code> <i>{connection string}</i>	<p>Stores the parameters for connecting to a proxy server that is used by the updater component (Dr.Web Updater) when it is connecting to Dr.Web updates servers (for example, if direct connections to external servers are prohibited by your network security policies).</p> <p>If the parameter value is not specified, the proxy server is not used.</p> <p>Allowed values:</p> <p><code><connection string></code> is the proxy server connection string. The string has the following format (URL):</p> <pre>[<protocol>://] [<user>:<password>@] <host> : <port></pre> <p>where:</p> <ul style="list-style-type: none">• <code><protocol></code> is the utilized protocol type (in the current version, only <code>http</code> is available);• <code><user></code> is the username to connect to the proxy server;• <code><password></code> is the password to connect to the proxy server;• <code><host></code> is the host address of the proxy (IP address or domain name, i.e. FQDN);• <code><port></code> is the port to be used. <p>The parts URL <code><protocol></code> and <code><user>:<password></code> may be absent. The proxy server address <code><host>:<port></code> is mandatory.</p> <p>If the username or password contains the following characters: '@', '%' or ':', these characters must be changed to the following HEX codes: "%40", "%25" and "%3A", respectively.</p>



Parameter	Description
	<p>Examples:</p> <ol style="list-style-type: none">In the configuration file:<ul style="list-style-type: none">Connection to a the proxy server hosted at <i>proxyhost.company.org</i> using port <i>123</i>: Proxy = proxyhost.company.org:123Connection to the proxy server hosted at <i>10.26.127.0</i> using port <i>3336</i> over HTTP protocol as user "legaluser" with the password "passw": Proxy = http://legaluser:passw@10.26.127.0:3336Connection to the proxy server hosted at <i>10.26.127.0</i>, using port <i>3336</i>, username "user@company.com", password "passw%123": Proxy = user%40company.com:passw%25123%3A@10.26.127.0:3336Setting the same values using the <code>drweb-ctl cfset</code> command:<pre data-bbox="635 824 1449 1059"># 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</pre> <p>Default value: <i>(not set)</i></p>
ExcludedFiles <i>{file name}</i>	<p>Defines the name of the file that will not be updated by the Dr.Web Updater component.</p> <p>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).</p> <p>Example: Add to the list the following files: <i>123.vdb</i> and <i>456.dws</i>.</p> <ol style="list-style-type: none">Adding of values to the configuration file.<ul style="list-style-type: none">Two values in one string:<pre data-bbox="635 1529 1449 1630">[Update] ExcludedFiles = "123.vdb", "456.dws"</pre>Two strings (one value per a string):<pre data-bbox="635 1693 1449 1827">[Update] ExcludedFiles = 123.vdb ExcludedFiles = 456.dws</pre>



Parameter	Description
	<p>2. Adding values via the <code>drweb-ctl cfset</code> command:</p> <pre># drweb-ctl cfset Update.ExcludedFiles -a 123.vdb # drweb-ctl cfset Update.ExcludedFiles -a 456.dws</pre> <p>Default value: <code>drweb32.lst</code></p>
<code>NetworkTimeout</code> <i>{time interval}</i>	<p>A time-out period imposed on the network-related operations of the updater component during the updating process.</p> <p>This parameter is used when a connection is temporarily lost: if the connection is established again before the time-out expires, the interrupted updating process will be continued.</p> <p>Specifying the time-out value larger than 75s has no effect as the connection is closed by TCP timeout.</p> <p>Minimum value: 5s.</p> <p>Default value: 60s</p>
<code>BaseDrlDir</code> <i>{path to directory}</i>	<p>Defines a path to directory that contains files used for connection to update servers of a standard update zone, which are used by the update component for updating virus databases and the scan engine.</p> <p>Default value: <code><var_dir>/drl/bases</code>.</p> <ul style="list-style-type: none">• For GNU/Linux: <code>/var/opt/drweb.com/drl/bases</code>.• For FreeBSD: <code>/var/drweb.com/drl/bases</code>
<code>BaseCustomDrlDir</code> <i>{path to directory}</i>	<p>Defines a path to directory that contains files used for connection to a special “customized” update zone, which are used by the for updating virus databases and the scan engine.</p> <p>If in the directory defined in parameter, is a non-empty signed server list file (<code>.drl</code> file), the update is performed only from these servers, and the main zone servers (see above) are not used to update the virus databases and the scan engine.</p> <p>Default value: <code><var_dir>/custom-drl/bases</code>.</p> <ul style="list-style-type: none">• For GNU/Linux: <code>/var/opt/drweb.com/custom-drl/bases</code>.• For FreeBSD: <code>/var/drweb.com/custom-drl/bases</code>
<code>BaseUpdateEnabled</code> <i>{Boolean}</i>	<p>Indicator that shows whether or not updating of virus databases and the scan engine is allowed.</p> <p>Allowed values:</p> <ul style="list-style-type: none">• <code>Yes</code>—updating is allowed and will be performed;• <code>No</code>—updating is not allowed and will not be performed. <p>Default value: <code>Yes</code></p>



Parameter	Description
<code>VersionDrlDir</code> <i>{path to directory}</i>	<p>Defines a path to directory that contains files used for connection to servers, which are used for updating Dr.Web for UNIX Internet Gateways versions.</p> <p>Default value: <code><var_dir>/drl/version</code>.</p> <ul style="list-style-type: none">• For GNU/Linux: <code>/var/opt/drweb.com/drl/version</code>.• For FreeBSD: <code>/var/drweb.com/drl/version</code>
<code>VersionUpdateEnabled</code> <i>{Boolean}</i>	<p>Indicator that shows whether or not updating of Dr.Web for UNIX Internet Gateways component version is allowed.</p> <p>Allowed values:</p> <ul style="list-style-type: none">• <code>Yes</code>—updating is allowed and will be performed;• <code>No</code>—updating is not allowed and will not be performed. <p>Default value: <code>Yes</code></p>
<code>DwsCustomDrlPath</code> <i>{path to file}</i>	<p>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.</p> <p>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.</p> <p>Default value: <code><var_dir>/drl/dws/custom.drl</code>.</p> <ul style="list-style-type: none">• For GNU/Linux: <code>/var/opt/drweb.com/drl/dws/custom.drl</code>.• For FreeBSD: <code>/var/drweb.com/drl/dws/custom.drl</code>
<code>DwsDrlDir</code> <i>{path to directory}</i>	<p>Defines a path the directory which contains the files to connect to servers of a standard update zone, which are used for updating database of web resource categories.</p> <p>Default value: <code><var_dir>/drl/dws</code>.</p> <ul style="list-style-type: none">• For GNU/Linux: <code>/var/opt/drweb.com/drl/dws</code>.• For FreeBSD: <code>/var/drweb.com/drl/dws</code>
<code>DwsCustomDrlDir</code> <i>{path to directory}</i>	<p>Defines a path the directory which contains the files to connect to servers of a special “customized” update zone, which are used for updating database of web resource categories.</p> <p>If in the directory defined in parameter, is a non-empty signed server list file (.drl file), the update is performed only from these servers, and the main zone servers (see above) are not used to update the databases of web resource categories.</p> <p>Default value: <code><var_dir>/custom-drl/dws</code>.</p> <ul style="list-style-type: none">• For GNU/Linux: <code>/var/opt/drweb.com/custom-drl/dws</code>.• For FreeBSD: <code>/var/drweb.com/custom-drl/dws</code>



Parameter	Description
DwsUpdateEnabled <i>{Boolean}</i>	Indicator that shows whether or not updating of database of web resource categories is allowed. Allowed values: <ul style="list-style-type: none">• Yes—updating is allowed and will be performed;• No—update is not allowed and will not be performed. Default value: Yes
AntispamDrlDir <i>{path to directory}</i>	The parameter is not used. Default value: <code><var_dir>/drl/antispam</code> . <ul style="list-style-type: none">• For GNU/Linux: <code>/var/opt/drweb.com/drl/antispam</code>.• For FreeBSD: <code>/var/drweb.com/drl/antispam</code>
AntispamCustomDrlDir <i>{path to directory}</i>	The parameter is not used. Default value: <code><var_dir>/custom-drl/antispam</code> . <ul style="list-style-type: none">• For GNU/Linux: <code>/var/opt/drweb.com/custom-drl/antispam</code>.• For FreeBSD: <code>/var/drweb.com/custom-drl/antispam</code>
AntispamUpdateEnabled <i>{Boolean}</i>	The parameter is not used. Default value: No
BackupDir <i>{path to directory}</i>	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: <code>/tmp/update-backup</code>
MaxBackups <i>{integer}</i>	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
IdleTimeLimit <i>{time interval}</i>	Maximum idle time for the component. When the specified period of time expires, the component shuts down. The component is launched upon the next update by schedule or an explicit <code>drweb-ctl update [--local-cloud]</code> command. When the update is completed, it is waiting for the specified time interval, and if there are no new requests (including interaction with the cloud if <code>UseLocalCloud = Yes</code>), then it shuts down until the next update attempt. Acceptable values: from 10 seconds (10s) to 30 days (30d) inclusive. If the <code>None</code> value is set, the component will functionate eternally; the <code>SIGTERM</code> signal will not be sent if the components goes idle.



Parameter	Description
	Default value: 30s
UseLocalCloud <i>{Boolean}</i>	<p>Interact with a local cloud of Dr.Web for UNIX products via the Dr.Web MeshD component for the update synchronization (send updates to the cloud, get updates from the cloud) in addition to the Dr.Web update servers.</p> <p>Allowed values:</p> <ul style="list-style-type: none">• No—update using Dr.Web update servers only. Update synchronization with the cloud is disabled, but can be performed explicitly, by the <code>drweb-ctl update --local-cloud</code> command;• Yes—update synchronization on the host with a local cloud (getting updates from the cloud if available, sending updates to the cloud, if the updates on the host are newer). <p>Default value: Yes</p>



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 [centralized protection](#) server (for example, to Dr.Web Enterprise Server).

When Dr.Web for UNIX Internet Gateways is connected to the centralized protection server Dr.Web ES Agent, the license [key file](#) are synchronized according to the key files stored on the centralized protection server. Moreover, Dr.Web ES Agent sends statistics on virus events, the list of running components and their status to the centralized protection server.

Dr.Web ES Agent also updates virus databases of Dr.Web for UNIX Internet Gateways directly from the connected centralized protection server bypassing the update component [Dr.Web Updater](#).

Operating Principles

Dr.Web ES Agent establishes connection to the centralized 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 centralized protection server also performs a role of an internal update server on the network, as it stores up-to-date virus databases, (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 centralized 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).

Dr.Web ES Agent collects and sends the server statistics on detected threats and applied actions.

To connect Dr.Web ES Agent to the centralized protection server, the password and identifier of the host ("station" in terms of the Centralized 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 centralized 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 centralized 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: 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: 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 the centralized 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 get 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 can be found in the [ESAgent] section of the integrated [configuration file](#) of Dr.Web for UNIX Internet Gateways.



The section contains the following parameters:

Parameter	Description
LogLevel <i>{logging level}</i>	<p>Logging level of the component.</p> <p>If the parameter value is not specified, the <code>DefaultLogLevel</code> parameter value from the [Root] section is used.</p> <p>Default value: <code>Notice</code></p>
Log <i>{log type}</i>	<p>Logging method of the component.</p> <p>Default value: <code>Auto</code></p>
ExePath <i>{path to file}</i>	<p>Executable path to the component.</p> <p>Default value: <code><opt_dir>/bin/drweb-esagent</code>.</p> <ul style="list-style-type: none">• For GNU/Linux: <code>/opt/drweb.com/bin/drweb-esagent</code>.• For FreeBSD: <code>/usr/local/libexec/drweb.com/bin/drweb-esagent</code>
DebugIpc <i>{Boolean}</i>	<p>Write IPC messages to the debug log <code>LogLevel = DEBUG</code> (interaction of Dr.Web ES Agent and the Dr.Web ConfigD configuration daemon).</p> <p>Default value: <code>No</code></p>
MobileMode <i>{On Off Auto}</i>	<p>Enable/disable the mobile mode when connected to a centralized protection server.</p> <p>Allowed values:</p> <ul style="list-style-type: none">• <code>On</code>— use the mobile mode if it is allowed by the centralized protection server (that is, perform updates from update servers of Doctor Web via Dr.Web Updater);• <code>Off</code>—do not use the mobile mode and continue operation in the centralized protection mode (updates are always received from the centralized protection server);• <code>Auto</code>—use the mobile mode, if allowed by the centralized protection server, and perform updates both from update servers of Doctor Web via Dr.Web Updater and from the centralized protection server, depending on which connection is available and which connection quality is higher. <p>Note that behavior of this parameter depends on server permissions: if the mobile mode is not allowed on the server in use, this parameter has no effect.</p> <p>Default value: <code>Auto</code></p>
Discovery <i>{On Off}</i>	<p>Enable/disable for the agent receiving <i>discovery</i> requests from the network inspector built in the centralized protection server (<i>discovery</i> requests are used by the inspector to check the structure and state of the anti-virus network).</p>



Parameter	Description
	<p>Allowed values:</p> <ul style="list-style-type: none">• <code>On</code>—enable receiving and processing <i>discovery</i> requests;• <code>Off</code>—disable receiving and processing <i>discovery</i> requests. <p>Note that this parameter has higher priority than settings of the centralized protection server: if the parameter value is set to <code>Off</code>, the agent does not receive discovery requests even if this option is enabled on the server.</p> <p>Default value: <code>On</code></p>
<code>UpdatePlatform</code> <i>{platform name}</i>	<p>Enable/disable for the agent receiving updates for the scan engine from the centralized protection server. The scan engine was developed for the indicated platform, where the <i>platform name</i> is a string, which contains the platform name.</p> <p>Allowed values:</p> <ul style="list-style-type: none">• for GNU/Linux: <code>unix-linux-32</code>, <code>unix-linux-64</code>, <code>unix-linux-mips</code>;• for FreeBSD: <code>unix-freebsd-32</code>, <code>unix-freebsd-64</code>;• for Darwin: <code>unix-darwin-32</code>, <code>unix-darwin-64</code> <div style="background-color: #fff9c4; padding: 10px; border: 1px solid #ccc;"><p> It is strongly recommended to change the parameter value only if you are sure it is required.</p></div> <p>Default value: <i>Depends on the platform currently being used</i></p>
<code>SrvMsgAutoremove</code> <i>{integer}</i>	<p>The storage period after which the messages from the centralized protection server are removed automatically.</p> <p>Allowed values: from 1 week (<code>1w</code>) to 365 days (<code>365d</code>). The storage period is specified as an integer with a suffix (<code>s</code>, <code>m</code>, <code>h</code>, <code>d</code>, <code>w</code>).</p> <p>Default value: <code>1w</code></p>



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 an interface to manage Dr.Web for UNIX Internet Gateways.

Besides managing Dr.Web for UNIX Internet Gateways through the Dr.Web 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.

For details about the HTTP API provided by Dr.Web HTTPD, refer to the [corresponding section](#).

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. Also, a user personal authorization certificate signed by a certificate authority certificate that is trusted by Dr.Web HTTPD can be used for automatic client authorization when connecting to Dr.Web HTTPD.

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 without impeding their operation.

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 secure HTTPS protocol is used for managing Dr.Web for UNIX Internet Gateways.



It is not mandatory to install Dr.Web management web interface for the proper functioning of Dr.Web for UNIX Internet Gateways. It can be missing. This is why the corresponding block is 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](#) component for file scanning, and to other components. These commands are based on those that were received through the provided HTTP API.



If the management web interface of Dr.Web for UNIX Internet Gateways, which uses Dr.Web HTTPD, is included in Dr.Web for UNIX Internet Gateways, it is described in the corresponding [section](#).

If the Dr.Web's management web interface is not included in Dr.Web for UNIX Internet Gateways, you can connect any external management interface that uses the HTTP API by Dr.Web HTTPD for interaction (described in the section [Description of the HTTP API](#)).

Command-Line Arguments

To run Dr.Web HTTPD, type the following command in the command line:

```
$ <opt_dir>/bin/drweb-httpd [<options>]
```

Dr.Web HTTPD can process the following options:

Parameter	Description
<code>--help</code>	Function: Output short help information about command-line parameters to the console or to the terminal emulator and to exit upon completion. Short form: <code>-h</code> Arguments: None.
<code>--version</code>	Function: Output information about the version of this component to the console or to the terminal emulator and to exit after completion. Short form: <code>-v</code> 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 get 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 can be found in the [HTTPD] section of the integrated [configuration file](#) of Dr.Web for UNIX Internet Gateways.

The section contains the following parameters:

Parameter	Description
LogLevel <i>{logging level}</i>	<p>Logging level of the component.</p> <p>If the parameter value is not specified, the <code>DefaultLogLevel</code> parameter value from the [Root] section is used.</p> <p>Default value: <code>Notice</code></p>
Log <i>{log type}</i>	<p>Logging method of the component.</p> <p>Default value: <code>Auto</code></p>
ExePath <i>{path to file}</i>	<p>Executable path to the component.</p> <p>Default value: <code><opt_dir>/bin/drweb-httpd</code>.</p> <ul style="list-style-type: none">• For GNU/Linux: <code>/opt/drweb.com/bin/drweb-httpd</code>.• For FreeBSD: <code>/usr/local/libexec/drweb.com/bin/drweb-httpd</code>
Start <i>{Boolean}</i>	<p>Launch/do not launch the component by the Dr.Web ConfigD configuration daemon.</p> <p>When you specify the <code>Yes</code> value for this parameter, it the configuration daemon will start the component immediately; and when you specify the <code>No</code> value, the configuration daemon will terminate the component immediately.</p> <p>Default value: <i>It depends on whether product management interface is installed.</i></p>
AdminListen <i>{address, ...}</i>	<p>List of network sockets (every network socket consists of <code><IP address>:<port></code>) on which Dr.Web HTTPD is listening for connections (via HTTPS) from clients that have administrative privileges. These sockets are used both for connecting to the managing web interface (if the web interface is installed) and for access to the HTTP API.</p> <p>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).</p> <p>Example: Add sockets <code>192.168.0.1:1234</code> and <code>10.20.30.45:5678</code> to the list.</p>



Parameter	Description
	<p>1. Adding values to the configuration file.</p> <ul style="list-style-type: none">• Two values in a line: <pre data-bbox="620 342 1449 479">[HTTPD] AdminListen = "192.168.0.1:1234", "10.20.30.45:5678"</pre> <ul style="list-style-type: none">• Two lines (a value per line): <pre data-bbox="620 539 1449 676">[HTTPD] AdminListen = 192.168.0.1:1234 AdminListen = 10.20.30.45:5678</pre> <p>2. Adding values via the <code>drweb-ctl cfset command</code>:</p> <pre data-bbox="620 739 1449 907"># drweb-ctl cfset HTTPD.AdminListen -a 192.168.0.1:1234 # drweb-ctl cfset HTTPD.AdminListen -a 10.20.30.45:5678</pre> <p>If no value is specified, it is impossible to use the HTTP API and the web interface (if it is installed).</p> <p>Default value: <code>127.0.0.1:4443</code></p>
<code>PublicListen</code> <i>{address, ...}</i>	<p>List of network sockets (every network socket consists of <code><IP address>:<port></code>) on which Dr.Web HTTPD is listening for connections (via HTTP) from clients with limited privileges.</p> <p>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).</p> <p>Example: Add sockets <code>192.168.0.1:1234</code> and <code>10.20.30.45:5678</code> to the list.</p> <p>1. Adding values to the configuration file.</p> <ul style="list-style-type: none">• Two values in a line: <pre data-bbox="620 1496 1449 1632">[HTTPD] PublicListen = "192.168.0.1:1234", "10.20.30.45:5678"</pre> <ul style="list-style-type: none">• Two lines (one value per line): <pre data-bbox="620 1693 1449 1830">[HTTPD] PublicListen = 192.168.0.1:1234 PublicListen = 10.20.30.45:5678</pre>



Parameter	Description
	<p>2. Adding values via the <code>drweb-ctl cfset</code> command:</p> <pre data-bbox="624 320 1449 488"># drweb-ctl cfset HTTPD.PublicListen -a 192.168.0.1:1234 # drweb-ctl cfset HTTPD.PublicListen -a 10.20.30.45:5678</pre> <p>At these addresses (sockets) you cannot access the full scope of the HTTP API commands or access the managing web interface.</p> <p>Default value: <i>(not set)</i></p>
AdminSslCertificate <i>{path to file}</i>	<p>Path to the server certificate file used by the web interface server for communication with clients that establish connections to an administrative socket via HTTPS.</p> <p>This file is generated automatically during the installation of the component.</p> <p>Please note that the certificate file and the private key file (which is specified by a parameter described below) must form a matching pair.</p> <p>Default value: <code><etc_dir>/certs/serv.crt</code>.</p> <ul style="list-style-type: none">• For GNU/Linux: <code>/etc/opt/drweb.com/certs/serv.crt</code>.• For FreeBSD: <code>/usr/local/etc/drweb.com/certs/serv.crt</code>
AdminSslKey <i>{path to file}</i>	<p>Path to private key file used by the web interface server for communication with clients that establish connections to an administrative socket via HTTPS.</p> <p>This file is generated automatically during the installation of the component.</p> <p>Please note that the certificate file (which is specified by the previous discussed parameter) and the private key file must form a matching pair.</p> <p>Default value: <code><etc_dir>/certs/serv.key</code>.</p> <ul style="list-style-type: none">• For GNU/Linux: <code>/etc/opt/drweb.com/certs/serv.key</code>.• For FreeBSD: <code>/usr/local/etc/drweb.com/certs/serv.key</code>
AdminSslCA <i>{path to file}</i>	<p>Path to a certificate file that acts as a trusted Certification Authority (CA) certificate for checking the certificates provided by the clients who are connecting to an administrative socket via HTTPS.</p> <p>If the client's certificate is signed with the certificate specified in this parameter, this client will not need to enter the login/password pair for authentication. Moreover, the login/password Authentication is prohibited for clients that use client certificates signed with the certificate set in this parameter.</p>



Parameter	Description
	<p>The client that passed this certificate-based authentication is always treated as a superuser (<i>root</i>).</p> <p>Default value: <i>(not set)</i></p>
<code>WebconsoleRoot</code> <i>{path to directory}</i>	<p>Path to the directory with the files used by the management web interface if it is installed (similar to the <code>htdocs</code> directory of an Apache HTTP Server).</p> <p>Default value: <code><opt_dir>/share/drweb-httpd/webconsole</code>.</p> <ul style="list-style-type: none">• For GNU/Linux: <code>/opt/drweb.com/share/drweb-httpd/webconsole</code>.• For FreeBSD: <code>/usr/local/libexec/drweb.com/bin/drweb-httpd/webconsole</code>
<code>AccessLogPath</code> <i>{path to file}</i>	<p>Path to the file where all HTTP/HTTPS requests from clients to the web interface server are registered.</p> <p>If not specified, HTTP/HTTPS requests are not logged to a file.</p> <p>Default value: <i>(not set)</i></p>

Description of the HTTP API

In this section

- [General Information](#)
- [User authentication and authorization](#)
- [Managing Dr.Web for UNIX Internet Gateways](#)
- [Managing the list of threats](#)
- [Managing Quarantine](#)
- [Examples of using the HTTP API](#)

1. General Information

The HTTP API is provided as a means to control and manage Dr.Web for UNIX Internet Gateways via the HTTP protocol (to ensure security, the HTTPS protocol is used).

Version 1.0 of the HTTP protocol is used. The API uses the standard methods of the HTTP protocol: `GET` and `POST`. All data is transferred in the form of JSON objects, except as otherwise specified. If you are sending a JSON object in the body of your HTTP `POST` request, use the `Content-Type: application/json` header with `application/json` as its value.



The format of an HTTP response to an HTTP request

- In responses to all requests, except as otherwise specified, JSON objects are returned. If an error occurs while processing a request, an [Error](#) JSON will be returned.
- If a JSON object sent as a response has a field of an Array type, but this array does not contain any elements, this field will be omitted in the response from the server.
- In all responses, except as otherwise specified, the `Content-Type:` header field has the `application/json` value.
- If the client requests an endpoint that does not exist, an [Error](#) JSON object with `EC_UNEXPECTED_MESSAGE` in the `code` field will be returned.
- If SCS (*Secure Cookie Sessions for HTTP*) is used (see [below](#)), the responses contain the SCS *cookie*.

Encoding of strings in JSON objects

- The strings are transmitted in UTF-8 encoding (without BOM). Symbols that are not part of the ASCII table are not escaped with sequences like `\uXXXX` in the outgoing JSON strings, but are transmitted in UTF-8 encoding.
- Strings in the incoming JSON objects may contain both UTF-8 encoded symbols and escaped sequences like `\uXXXX`.

General restrictions of data transmission

- In POST-requests with JSON objects in the body any symbols complying with [RFC 7159](#) are allowed.
- In GET-requests any symbols complying with [RFC 1945](#) are allowed in the URI.
- Symbols complying with [RFC 1945](#) can be used in any other part of the request (either in the headers or in the body).

2. User authentication and authorization

To start using the API you should be authenticated by the server. Two means of authorization are provided.

1. [Using SCS](#), according to [RFC 6896](#).
2. [Using clients' SSL certificates](#) that are signed with a special certificate which Dr.Web HTTPD regards as a trusted CA's certificate. In this case the client is treated as if the client had successfully input root credentials to get authorization (X.509 client certificates are used).

If a SCS is used, the cookies confirming the authentication are transferred in the headers:

`Cookie:` in the request and `Set-Cookie:` in the response.

When authorizing with an SSL certificate, no cookies are used.



When authorizing with SCS, using the API starts with sending the `login` command. If this command is executed successfully, an SCS cookie is sent to the client in the response. When authorizing with a client certificate, you do not need to run the `login` command. If you try to execute it, an [Error](#) JSON object will be returned in the response.

2.1. Via specifying login and password (SCS)

User authentication and authorization commands:

API command	Description
<code>login</code>	<p>Action: Authenticate the client based on the specified user name and password and authorize the client to use the HTTP API commands. If the authentication is successful, an SCS cookie will be returned.</p> <p>URI: <code>/api/10.2/login</code></p> <p>HTTP method: POST</p> <p>Input parameters: AuthOptions object</p> <p>Result of successful execution: an empty object, SCS cookie</p>
<code>logout</code>	<p>Action: Revoke the provided SCS cookie. After that, in response to any HTTP API call that contains the revoked SCS cookie an Error object will be returned that will contain the <code>EC_NOT_AUTHORIZED</code> error code.</p> <p>URI: <code>/api/10.2/logout</code></p> <p>HTTP method: GET</p> <p>Input parameters: SCS cookie</p> <p>Result of successful execution: an empty object</p>
<code>whoami</code>	<p>Action: Display the name of the authenticated user.</p> <p>URI: <code>/api/10.2/whoami</code></p> <p>HTTP method: GET</p> <p>Input parameters: (SCS cookie)*</p> <p>Result of successful execution: whoami object, (SCS cookie)</p>

*) Here and below the SCS cookie is given in parentheses, because sending/receiving it is required only if authorization via SCS is used.



The `login` и `logout` commands are used only when authenticating with SCS.



Description of used objects

1) `AuthOptions`—an object that contains the login data of a user that needs to be authenticated and authorized to use the full HTTP API:

```
{
  "user": string, //User name
  "password": string //User's password
}
```



You can specify a user who is a member of the admin group (`sudoers` in Debian and Ubuntu, `wheel` in CentOS and Fedora, `astra-admin` in Astra Linux, etc). If the user is not a member of the admin group, the `EC_NOT_AUTHORIZED` error will be returned in the response.

2) `whoami`—an object that contains the name of the user that was authorized to use the HTTP API:

```
{
  "whoami" :
  {
    "user": string //User name
  }
}
```

3) `Error`—an object that contains Information about an error that has occurred:

```
{
  "error" :
  {
    "code" : string, //A string specifying an error code that looks like
    EC_XXX
    *"what": string //Description of the error
  }
}
```

Asterisk-marked parameter is optional.



The `Error` JSON object that is returned in response to a HTTP API command if an error occurs while processing the request, has a `code` field that contains not a numeric error code, but an internal string-type code that is used by the components of Dr.Web for UNIX Internet Gateways. This code is a string that looks like `EC_XXX`. To find out the corresponding numeric code and to get detailed information about the error please refer to the [Known Errors](#) section (in the Appendix F to the Administrator Manual).



2.2. Authentication using a personal certificate

Authentication with an SSL certificate supposes using the personal certificate is signed by a certificate authority certificate specified in the Dr.Web HTTPD settings as trusted. If you are authenticated with a certificate, all your requests are considered a made on behalf of the *root* user.

To authorize with a personal user certificate

1. Create a personal certificate signed by a certificate authority certificate.
2. In the Dr.Web HTTPD [settings](#) (parameter `AdminSSLCA`), specify a path to the authority certificate by which your personal certificate is signed.
3. Each time you connect to Dr.Web HTTPD, use a signed certificate.

If necessary, refer to the [Appendix E. Generating SSL certificates](#) section.

3. Managing Dr.Web for UNIX Internet Gateways

API commands for viewing and modifying the current values of configuration parameters:

API command	Description
<i>Configuration management commands</i>	
<code>get_lexmap</code>	Action: Get the parameter values of the current configuration (called a "lexical map" of parameters here). URI: <code>/api/10.2/get_lexmap</code> HTTP method: GET Input parameters: (SCS cookie) Result of successful execution: LexMaps object, (SCS cookie)
<code>set_lexmap</code>	Action: Set or reset (to defaults) the specified parameters of the current configuration (sent as a "lexical map" of parameters). URI: <code>/api/10.2/set_lexmap</code> HTTP method: POST Input parameters: (SCS cookie), LexMap object Result of successful execution: SetOptionResult object, (SCS cookie)
<i>Updating commands</i>	
<code>start_update</code>	Action: Launch update.



API command	Description
	<p>URI: <code>/api/10.2/start_update</code></p> <p>HTTP method: POST</p> <p>Input parameters: (SCS cookie)</p> <p>Result of successful execution: the StartUpdate object, (SCS cookie)</p>
<code>stop_update</code>	<p>Action: Stop active updating process.</p> <p>URI: <code>/api/10.2/stop_update</code></p> <p>HTTP method: POST</p> <p>Input parameters: (SCS cookie)</p> <p>Result of successful execution: an empty object, (SCS cookie)</p>
<code>baseinfo</code>	<p>Action: View the information on the downloaded viral bases.</p> <p>URI: <code>/api/10.2/baseinfo</code></p> <p>HTTP method: GET</p> <p>Input parameters: (SCS cookie)</p> <p>Result of successful execution: the BaseInfoResult object including the VirusBaseInfo object (SCS cookie)</p>
<i>Licence management commands</i>	
<code>install_license</code>	<p>Action: Install the specified key file.</p> <p>URI: <code>/api/10.2/install_license</code></p> <p>HTTP method: POST</p> <p>Input parameters: (SCS cookie), key file body (or an archive with the key file)</p> <p>Result of successful execution: an empty object, (SCS cookie)</p>
<i>Commands for connection to the centralized protection server</i>	
<code>esconnect</code>	<p>Action: Enable the centralized protection mode.</p> <p>URI: <code>/api/10.2/esconnect</code></p> <p>HTTP method: POST</p> <p>Input parameters: (SCS cookie), the ESConnection object</p>



API command	Description
	Result of successful execution: an empty object, (SCS cookie)
esdisconnect	Action: Disable the centralized protection mode. URI: /api/10.2/esdisconnect HTTP method: POST Input parameters: (SCS cookie) Result of successful execution: an empty object, (SCS cookie)

Configuration of the product components is returned and set as a so called *lexical map*, i.e. as a sequence of parameter-value pairs. A [LexMaps](#) object always contains three enclosed [LexMap](#) objects):

- *active*—the current values of the parameters;
- *hardcoded*—default values automatically assigned to the parameters whose values are missing or invalid;
- *master*—the values of configuration parameters set by the client.



The `get_lexmap` command always returns all three sets of configuration parameter values for all the components that can be included in Dr.Web for UNIX Internet Gateways, not only for those that are actually installed and running.

Description of JSON objects

1) [LexMaps](#)—an object that contains the active, the default and the user-set lexical maps of parameter values:

```
{
  "active": LexMap, //Active (current) values of configuration parameters
  "hardcoded": LexMap, //Default values of configuration parameters
  "master": LexMap //Configuration parameter values set
  //by the user
}
```

Each of these fields is a [LexMap](#) object, which in turn contains an array of [LexOption](#) objects.

2) [LexMap](#)—an object that contains a lexical map of parameters:

```
{
  "option": LexOption[] //Array of configuration options
}
```

3) [LexOption](#)—an object that contains a single parameter or a section of the configuration (a section is a group of parameters):



```
{
  "key": string, //Name of the option (configuration parameter/section)
  *"value": LexValue, //If this option is a single parameter
  *"map": LexMap //If this option is a section
}
```

Asterisk-marked parameters are optional.

The [LexOption](#) object represents a section or a single parameter of the Dr.Web for UNIX Internet Gateways configuration. It always has a `key` field that corresponds to the name of the section or to the name of the single parameter. Besides this, depending on what this object describes—a single parameter or a section—the object also contains either a `value` field (in case it describes a single parameter), or a `map` field (in case it describes a section). Any section is also an object of the [LexMap](#) type; whereas the value of a single parameter is an object of the [LexValue](#) type that contains an `item` field specifying the parameter value in the string format.

- 4) [LexValue](#)—an object that contains the array of values assigned to a parameter:

```
{
  "item": string[] //Array of parameter values
}
```

As its input, the `set_lexmap` command accepts a [LexMap](#) object, which must contain all the parameters whose values you want to change to new ones or want to reset to their defaults. Those parameters that you want to reset to their default values must not contain the `value` field. Parameters that are not mentioned in the lexical map given by you in the `set_lexmap` command will not be changed. As a result of its execution, the `set_lexmap` command returns a [SetOptionResult](#) object that will contain the results of changing every parameter that was specified in your command.

- 5) [SetOptionResult](#)—an object with an `item` field that contains an array of results of parameter changes:

```
{
  "item": SetOptionResultItem[] //Array of results
}
```

The object contains an array of [SetOptionResultItem](#) objects that describe the results of making changes to every parameter specified in your command.

- 6) [SetOptionResultItem](#)—an object that contains information about a making a change to a value of some parameter:

```
{
  "option": string, //Name of the parameter
  "result": string, //Result of changing the value (error code)
  *"lower_limit": string, //The lowest permitted value
  *"upper_limit": string //The highest permitted value
}
```

Asterisk-marked parameters are optional.



The `option` field contains the name of the parameter to which your action was applied, and the `result` field contains the result of an attempt to change the value of this parameter. If the new value was assigned successfully to a parameter, then this field will contain `EC_OK`. In case of an error (if this field is not equal to `EC_OK`), this object may optionally contain an `lower_limit` field and an `upper_limit` field that hold the maximum and the minimum permitted value for this parameter.

7) The `StartUpdate` object contains data on the started update process:

```
{
  "start_update":
  {
    "attempt_id" : number //Identifier of a launched updating process
  }
}
```

8) The `ESConnection` object contains data on the started update process:

```
{
  *"server": string,    //<Host address>:<port> (without the http/https
  prefix)
  "certificate": string, //Base64 server key
  *"newbie": boolean,   //False by default
  *"login": string,     //User name
  *"password": string   //Password
}
```

Asterisk-marked parameters are optional.

The parameters `login` and `password` are only specified if `newbie = true`. Before connecting, download the certificate file from the centralized protection server and execute the command:

```
$ cat certificate.pem |base64
```

The string obtained from executing this command is used as the parameter value for `certificate`.

9) The `BaseInfoResult` object contains data on the downloaded viral bases:

```
{
  "vdb_base_stamp" : number //Timestamp of the base
  "vdb_bases" : VirusBaseInfo[] //Detailed information upon the base
}
```

10) The `VirusBaseInfo` object contains the information on each virus base:

```
{
  "path" : string //Path to the base file
  "virus_records" : number //The number of records in the base
  "version" : number //Base version
  "timestamp" : number //Base timestamp
  "md5" : string //base MD5-hash
  "load_result" : string //The result of downloading the base (EC_OK if
  the base has been downloaded successfully)
```



```
  *"sha1" : string - base SHA1-hash  
}
```

Asterisk-marked parameter is optional.

4. Object Scanning

API commands for scanning objects:

API command	Description
<i>Data scanning (using the Dr.Web Network Checker component call)</i>	
scan_request	Action: Order of connection (<i>endpoint</i>) to scan data with the necessary parameters. URI: /api/10.2/scan_request HTTP method: POST Input parameters: (SCS cookie), the ScanOptions object Result of successful execution: the ScanEndpoint object, (SCS cookie)
scan_endpoint	Action: Launch of data scanning (for instance, file body) at the created <i>endpoint</i> connection. URI: /api/10.2/scan_endpoint/<endpoint> HTTP method: POST Input parameters: (SCS cookie), verifiable data Result of successful execution: the ScanResult object, (SCS cookie)
scan_path	Action: Scanning a file or a directory located on the path specified. URI: /api/10.2/scan_path HTTP method: POST Input parameters: (SCS cookie), ScanPathOptions object Result of successful execution: the ScanPathResult object, (SCS cookie)
scan_stat	Action: Viewing scan statistics. URI: /api/10.2/scan_stat HTTP method: GET Input parameters: (SCS cookie), statistics format (JSON or CSV)



API command	Description
	<p>Result of successful execution: the ScanStat object (if JSON format is selected), (SCS cookie)</p> <p>If CSV format is selected, a table whose columns correspond to the fields the ScanStat is returned.</p>

Description of JSON objects

1) ScanOptions is the object that contains parameters used for creating *endpoint* for file scanning:

```
{
  "scan_timeout_ms": number, //A time-out to scan one file, in ms
  "cure": boolean, //Apply cure to infected file
  "heuristic_analysis": boolean, //Use heuristic analysis
  "packer_max_level": number, //Maximum nesting level for packed objects
  "archive_max_level": number, //Maximum nesting level for archives
  "mail_max_level": number, //Maximum nesting level for email messaged
  "container_max_level": number, //Maximum nesting level for other compound
objects (containers)
  "max_compression_ratio": number, //Maximum a compression value
  "min_size_to_scan" : number, //Minimal size of an object to be scanned
  "max_size_to_scan" : number, //Maximum size of an object to be scanned
  "threat_hash" : boolean //Return SHA1 and SHA256 of all threats
}
```

2) ScanPathOptions is the object that contains parameters used for scanning a file or a directory located on the path specified:

```
{
  "path" : string //Absolute path to the file or the directory to be
scanned
  *"exclude_path" : string[] //List of the paths excluded from scanning
(it is allowed to use masks)
  *"scan_timeout_ms" : number //Scan timeout for an object
  *"archive_max_level" : number//Maximum nesting level for archived
objects
  *"packer_max_level" : number //Maximum nesting level for packed
objects
  *"mail_max_level" : number //Maximum nesting level for email messages
  *"container_max_level" : number//Maximum nesting level for other
compound objects (containters)
  *"max_compression_ratio" : number//Maximum compression value
  *"heuristic_analysis" : bool //Use heuristic analysis (true by
default)
  *"follow_symlinks" : bool //Follow symbolic links
  *"min_size_to_scan" : number //Minimal size of an object to be scanned
  *"max_size_to_scan" : number //Maximal size of an object to be scanned
  *"timeout_ms" : number - //Scan timeout for all objects
  *"threat_hash" : bool - //Return SHA1 and SHA256 of all threats
}
```



Asterisk-marked parameters are optional.

3) ScanPathResult is an object that contains results of the scanning of the object located on the path specified:

```
{
  ScanPathResult:
    "results": ScanResult[] //Scan results
    *"error": string //Error if the scanning process terminated (the
scanning timeout is expired, for instance)
}
```

Asterisk-marked parameter is optional.

If the scanning is successful, the error string is not included in the response.

4) ScanResult is an object that contains the results of scanning:

```
{
  ScanResult:
    "scan_report" : ScanReport //The information upon the threat found
    *"sha1" : string //The SHA1 hash of the threat
    *"sha256" : string //The SHA256 hash of the threat
}
```

Asterisk-marked parameters are optional.

5) ScanReport is an object that contains information about the file in which the threat was found:

```
{
  ScanReport:
  "object" : string //Name of the object scanned
    For a file //The absolute path, for a nested object - the name of the
file
    Always points to temporary file when calling scan_endpoint
  *"size" : number //Object size
  *"compressed_size" : number //Object size before extraction
  *"core_fingerprint" : string //Scan engine fingerprint
  *"packer" : string[] //The list of packers used to pack the object
  *"compression_ratio" : number //Archive compression ratio
  *"archive" : Archive //Information on the archive or container type, if
the object scanned was identified as an archive or a container
  *"virus" : Virus[] //Viruses detected in the objects (if found)
  *"item" : ScanReport[] //Reports on scanning of the nested objects (if
there were some)
  *"error" : string //Scanning error (if occurred)
  *"heuristic_analysis" : bool //Indicates if heuristic analysis was used
  *"cured" : bool //The object was cured
  *"cured_by_deletion" : bool //The object was deleted.
  *"new_path" : string //The new path to the object renamed when being cured
  *"user_time" : number //Type spent for syscalls when scanning
  *"system_time" : number //Time spent in the userspace
}
```

Asterisk-marked parameters are optional.



The fields `virus` and `error` may be absent if no threats are detected and no errors occur during the scan. To call `scan_endpoint`, the `scan_endpoint` field always specifies a temporary file, created by the Dr.Web Network Checker component in the local server file system and containing the data for scanning, sent in the body of the `scan_endpoint` request.

6) `ScanEndpoint` is an object that contains data on the *endpoint*, created for file scanning:

```
{
  "endpoint": string //Unique identifier of the created endpoint
}
```

The `endpoint` string, returned in the object body, is used to launch file scanning with the `scan_endpoint` command (part of URI).

7) `VirusInfo` is an object that contains information on the detected threat:

```
{
  "type": string, //Type of the detected threat
  "name": string //Name of the threat
}
```

The `type` field (threat type) is the string `SE_XXX`:

- `SE_KNOWN_VIRUS` is a known virus;
- `SE_VIRUS_MODIFICATION` is a modification of known malware;
- `SE_UNKNOWN_VIRUS` is an unknown virus (suspicious object);
- `SE_ADWARE` is adware;
- `SE_DIALER` is a dialer program;
- `SE_JOKE` is a joke program;
- `SE_RISKWARE` is a potentially dangerous program;
- `SE_HACKTOOL` is a hacktool.

8) `Archive` is an object that contains information on archives, packed objects, e-mail messaged and other containers:

```
{
  "type" : string - the type of the archive:
    "SE_ARCHIVE" - archive
    "SE_MAIL" - e-mail message
    "SE_CONTAINER" - other container
  "name" : string - archive format
}
```

9) `ScanStat` is an object that contains the scanning statistics:

```
{
  "origin": string //The application by the request of which the scanning
was initialized
  #Counters for infected objects:
  "known_virus": number //Number of objects infected by known viruses
}
```



```
"virus_modification": number //Number of objects infected by
modifications of known viruses
"unknown_virus": number //Number of objects infected by unknown
viruses
"adware": number //Number of objects with SE_ADWARE
"dialer": number //Number of objects with SE_DIALER
"joke": number //Number of objects with SE_JOKE
"riskware": number //Number of objects with SE_RISKWARE
"hacktool" : number //Number of objects with SE_HACKTOOL
"cured": number //Number of cured threats
"quarantined": number //Number of quarantined threats
"deleted": number //Number of deleted threats
}
```

5. Managing the list of threats

To manage the list of threats that have been detected during a scanning or by the file system monitor—SpIDer Guard—the following commands are available in the HTTP API:

API command	Description
threats	Action: List identifiers of all detected threats. URI: /api/10.2/threats/ HTTP method: GET Input parameters: (SCS cookie) Result of successful execution: Array of threat identifiers
threat_info	Action: Get information about a threat by its identifier—<threat ID>. URI: /api/10.2/threat_info/<threat ID> HTTP method: GET Input parameters: (SCS cookie) Result of successful execution: (SCS cookie), FileThreat object
cure_threat	Action: Try to cure a threat specified by its identifier—<threat ID>. URI: /api/10.2/cure_threat/<threat ID> HTTP method: POST Input parameters: (SCS cookie) Result of successful execution: (SCS cookie), an empty object



API command	Description
<code>delete_threat</code>	<p>Action: Delete the file that contains the threat specified by the threat identifier—<i><threat ID></i>.</p> <p>URI: <code>/api/10.2/delete_threat/<threat ID></code></p> <p>HTTP method: POST</p> <p>Input parameters: (SCS cookie)</p> <p>Result of successful execution: (SCS cookie), an empty object</p>
<code>ignore_threat</code>	<p>Action: Ignore a threat specified by its identifier—<i><threat ID></i>.</p> <p>URI: <code>/api/10.2/ignore_threat/<threat ID></code></p> <p>HTTP method: POST</p> <p>Input parameters: (SCS cookie)</p> <p>Result of successful execution: (SCS cookie), an empty object</p>
<code>quarantine_threat</code>	<p>Action: Quarantine a threat specified by its identifier—<i><threat ID></i>.</p> <p>URI: <code>/api/10.2/quarantine_threat/<threat ID></code></p> <p>HTTP method: POST</p> <p>Input parameters: (SCS cookie)</p> <p>Result of successful execution: (SCS cookie), an empty object</p>

Each threat found in the specified application has a unique numeric integer identifier *<threat ID>*. The list of all identifiers is returned by the `threats` command. In the `threat_info`, `cure_threat`, `delete_threat`, `ignore_threat`, and `quarantine_threat` commands only the identifiers returned by the `threats` command are allowed.

All information on the specified threat, including actions history, can be obtained using the `threat_info` request. The information is returned as the [FileThreat](#) object.

Description of JSON objects

1) `FileThreat` is an object that contains the following data:

```
{
  "threat_id": number, //Threat identifier
  "detection_time": UNIXTime, //Time when the threat was detected
  "report": ScanReport, //Report about scanning the file
  "stat": FileStat, //Information about the file
  "origin": string, //Name of the component that detected the threat
```



```
"origin_pid": number, //PID of the component that detected the threat
"task_id": number, //Identifier of the scanning task
//in the scan engine
"history": ActionResult[] //History of actions applied to the threat (an
array)
}
```

The `report` field contains the [ScanReport](#) object; the `stat` field contains a [FileStat](#) object, and the `history` field contains an array of [ActionResult](#) objects (the history of actions applied to the file).

2) `ScanReport` is an object that contains information about the file in which the threat was found:

```
{
"object": string, //File system object that contains the threat
"size": number, //Size (in bytes) of the file that contains the threat
"virus": VirusInfo[], //List of details about the found
//threats
*"error": string, //An error message
"heuristic_analysis": bool //Flag that shows whether heuristic
//analysis was used
}
```

Asterisk-marked parameter is optional.

The `virus` field is an array of [VirusInfo](#) objects containing information about all detected threats. The `error` field is only present if an error has occurred.

3) `FileStat` is an object that contains file statistics:

```
{
"dev": number, //Device containing the file
"ino": number, //The file inode number
*"size": number, //Size of the file
*"uid": User, //User ID of the file's owner
*"gid": Group, //Group ID of the owning group
*"mode": number, //The mode of access to the file
*"mtime": UNIXTime, //Date/time when the file was last modified
*"ctime": UNIXTime //Date/time when the file was created
*"rsrc_size": number, //
*"finder_info": string, //
*"ext_finder_info": string, //
*"uchg": string, //
*"volume_name": string, //Volume name
*"volume_root": string, //Root (mount point) of the volume
*"xattr": XAttr[] //Extended information about the file
}
```

Asterisk-marked parameters are optional.

The `xattr` field contains an array of `XAttr` objects. This object contains two *string*-type fields: `name` and `value`. The `uid` and `gid` fields contain a `User` and a `Group` object respectively, that have information about the file owner and the group owning the file respectively. These objects contain two fields each:



- uid (gid)—the numeric ID of the user (group);
- username (groupname)—name of the user (group) as a string.

4) `ActionResult` is an object that contains information on the action applied to the file and the result:

```
{
  "action": string, //The action applied
  "action_time": UNIXTime, //Date/time when the action was applied
  "result": string, //Result of applying the action
  "cure_report": ScanReport //Report about applying the action
}
```

The `cure_threat`, `delete_threat`, `ignore_threat` and `quarantine_threat` commands return an empty object when executed successfully. If the requested action fails, [Error](#) object is returned.

6. Managing Quarantine

To manage quarantined objects the following commands are available in the HTTP API:

API command	Description
<code>quarantine</code>	<p>Action: List identifiers of quarantined objects.</p> <p>URI: <code>/api/10.2/quarantine/</code></p> <p>HTTP method: GET</p> <p>Input parameters: (SCS cookie)</p> <p>Result of successful execution: (SCS cookie), array of QuarantineId objects (objects in quarantine)</p>
<code>qentry_info</code>	<p>Action: Get information about a quarantined object specified by its identifier—<code><entry ID></code>.</p> <p>URI: <code>/api/10.2/qentry_info/<entry ID></code></p> <p>HTTP method: GET</p> <p>Input parameters: (SCS cookie)</p> <p>Result of successful execution: (SCS cookie), QEntry object</p>
<code>cure_qentry</code>	<p>Action: Try to cure a quarantined object specified by its identifier—<code><entry ID></code>.</p> <p>URI: <code>/api/10.2/cure_qentry/<entry ID></code></p> <p>HTTP method: POST</p>



API command	Description
	Input parameters: (SCS cookie) Result of successful execution: (SCS cookie), an empty object
<code>delete_qentry</code>	Action: Delete a quarantined object specified by its identifier— <code><entry ID></code> . URI: <code>/api/10.2/delete_qentry/<entry ID></code> HTTP method: POST Input parameters: (SCS cookie) Result of successful execution: (SCS cookie), an empty object
<code>restore_qentry</code>	Action: Restore a quarantined object specified by its identifier— <code><entry ID></code> —to its original location. URI: <code>/api/10.2/restore_qentry/<entry ID></code> HTTP method: POST Input parameters: (SCS cookie) Result of successful execution: (SCS cookie), an empty object

Each quarantined object has a unique identifier. The list of identifiers represented as of [QuarantineId](#) is returned by the `quarantine` command. The identifier consists of two parts: `chunk_id` and `entry_id`.

Description of JSON objects

1) `QuarantineId` is an object that contains both parts of the two-part identifier of a quarantined object:

```
{
  "chunk_id": string,
  "entry_id": string
}
```

These two fields together make up the identifier of a quarantined object. To apply any action to a quarantined object with the help of the `qentry_info`, `cure_qentry`, `delete_qentry` or `restore_qentry` commands, you should specify the quarantined object general identifier—`<entry ID>`—written in the form of `<entry_id>@<chunk_id>`. The `qentry_info` command allows to get the detailed information about a quarantined object with a specified identifier. This command returns a [QEntry](#) object.

2) `QEntry` is an object that contains information about a quarantined object:



```
{
  "entry_id": string, //Parts of the identifier of
  *"chunk_id": string, //this quarantined object
  *"quarantine_dir": string, //Quarantine directory
  "restore_path": string, //path where the quarantined
  //object will be restored
  "creation_time": number, //Date/time of moving to quarantine
  //(in UNIX time)
  "report": ScanReport, //Report about scanning the object
  //(see ScanReport described above)
  "stat": FileStat, //Statistical information about the file
  //(see FileStat described above)
  *"history": QEntryOperation[], //History of operations performed on the
  object
  *"who": RemoteUser, //The remote owner of the file (if
  //the file was quarantined from a file server
  //storage)
  *"detection_time": number, //Date/time of detecting the threat
  *"origin": string, //Component that detected the threat
}
```

Asterisk-marked parameters are optional.

The `report` field contains a [ScanReport](#) object; the `stat` field contains a [FileStat](#) object, and the `history` field contains the history of actions applied to the isolated object. Each action entry is described by a [QEntryOperation](#) object. The optional `who` field contains information about the deleted user in the form of a [RemoteUser](#) object.

3) `QEntryOperation` is an object that contains data on operations applied to the quarantined object:

```
{
  "action": string, //Operation performed on the object
  //(see the possible values below)
  "action_time": number, //Date/time when the operation was performed (UNIX
  Time)
  "result": string, //Error when trying to perform the operation (a code
  //EC_XXX)
  *"restore_path": string, //path for restoring the quarantined object
  //(if action = "QENTRY_ACTION_RESTORE")
  *"rescan_report": ScanReport //Report about rescanning (if
  //action = "QENTRY_ACTION_RESCAN")
}
```

Asterisk-marked parameters are optional.

The `action` field can have the following values:

- `QENTRY_ACTION_DELETE` is an attempt to remove the quarantined object;
- `QENTRY_ACTION_RESTORE` is an attempt to restore the quarantined object;
- `QENTRY_ACTION_RESCAN` is an attempt to rescan the quarantined object;
- `QENTRY_ACTION_CURE` is an attempt to cure the quarantined object.

4) `RemoteUser` is an object that contains information about the remote user that owns the file (if the file was relocated to quarantine from the file server storage):



```
{
  *"ip": string, //IP-address of the user
  *"user": string, //User name
  *"domain": string //Domain of the user
}
```

Asterisk-marked parameters are optional.

The `cure_qentry`, `delete_qentry` and `restore_qentry` commands return an empty object if the command execution was successful. In case the requested operation on a quarantined object finished with an error (for instance, the file could not be restored), then, instead of an empty object, an [Error](#) object will be returned.

7. Examples of using the HTTP API

To test the work of the HTTP API, you can use the `curl` utility. The general format of an API call is given below.

```
$ curl https://<HTTPD.AdminListen>/<HTTP API URI> -k -X <HTTP method name>
[-H 'Content-Type: application/json' --data-binary '@<file of the JSON object>']
[-c <cookie file> [-b <cookie file>]] [> <file of the result>]
```

- the `-k` option specifies `curl` not to verify the SSL certificate;
- the `-X` options specifies the HTTP method used (GET or POST);
- the `-H` option is used to add the `Content-Type: application/json` header;
- the `--data-binary` (or `-d`) option is used to add a JSON object saved in a text file to the request;
- if an SCS is used to get authorization, the files that contain the sent and received SCS cookies need to be specified with the `-b` and `-c` parameters respectively.

For the detailed description of the `curl` options refer to the man page (perform `curl --help` or `man curl` command).

1. **Authenticate and authorize a client** by specifying a user name and a password (for an SCS).

An [AuthOptions](#) object in the JSON format must be written in advance to a file named `user.json`. For example:

```
{"user": "<username>", "password": "<passphrase>"}
```

Request:

```
$ curl https://127.0.0.1:4443/api/10.2/login -k -X POST -H 'Content-Type:
application/json' --data-binary '@user.json' -c cookie.file
```

Response:

```
HTTP/1.0 200 OK
Content-Type: application/json
```



```
Content-Length: 2
Set-Cookie:
DWTOKEN=6QXy4wn_JGov9A1GohWP_kvMK3dN6ccKegjNgKcmHpb_AqSrHg9cNX_yFJhxPDgr|
MTQ2Mjg3Mzg4NQ==|cWd4Ow==|GywBUVOhU4w2LF_BKT5frg==|
kR_rip5nrpxWjJ2dfZ7Xfmvi3rE=; Secure; HttpOnly; Max-Age: 900; Path=/
Pragma: no-cache

{}
```

The `Set-Cookie` header field contains an SCS cookie that should be used in all further requests to the HTTP API. The body of the response contains an empty object, if the authentication and authorization were successful. If the user has not been authorized, an [Error](#) object is returned:

```
HTTP/1.0 403 Forbidden
Content-Type: application/json
Content-Length: 35
Pragma: no-cache

{"error":{"code":"EC_AUTH_FAILED"}}
```

2. Get the information about the threat with *ID = 1*:

Request:

```
$ curl https://127.0.0.1:4443/api/10.2/threat_info/1 -k -X GET -c
cookie.file -b cookie.file
```

Response:

```
HTTP/1.0 200 OK
Content-Type: application/json
Content-Length: 574
Set-Cookie: DWTOKEN=<...>;
Secure; HttpOnly; Max-Age: 900; Path=/
Pragma: no-cache

{"threat_id":1,"detection_time":1462881660,
"report":{"object":"/sites/site1/eicar.com.txt","size":68,"packer":[],
"virus":[{"type":"SE_KNOWN_VIRUS","name":"EICAR Test File (NOT a
Virus!)"}]},
"heuristic_analysis":true,"core_fingerprint":"0D2DD5A869DAB7AE354153A4D5F7
0F45",
"item":[],"log":[],"user_time":0,"system_time":0},"stat":
{"dev":2049,"ino":898,
"size":68,"uid":{"uid":1000,"username":"user"},"gid":
{"gid":1000,"groupname":"user"},
"mode":33204,"mtime":1441028214,"ctime":1460738554,"xattr":[]},
"origin":"APP_COMMAND_LINE_TOOL","origin_pid":2726,"task_id":1,"history":
[]}
```

3. Move to quarantine the threat with *ID = 1*:

Request:



```
$ curl -v -c cookie.jar -b cookie.jar -k -X POST -H 'Content-Type:application/json' https://127.0.0.1:4443/api/10.2/quarantine_threat/1
```

Response:

```
HTTP/1.0 200 OK
Content-Type: application/json
Content-Length: 2
Set-Cookie: DWToken=<...>; Secure; HttpOnly; Max-Age: 900; Path=/
Pragma: no-cache

{}
```

4. View the information on the quarantined (isolated) object with the specified ID:

Request:

```
$ curl -v -k -X GET -c cookie.jar -b cookie.jar
https://127.0.0.1:4443/api/10.2/qentry_info/3070d3ce-7b6e-4143-9d9f-89ba3473a781@801:2108d
```

Response:

```
HTTP/1.0 200 OK
Content-Type: application/json
Content-Length: 781
Set-Cookie: DWToken=<...>; Secure; HttpOnly; Max-Age: 900; Path=/
Pragma: no-cache

{"entry_id":"3070d3ce-7b6e-4143-9d9f-89ba3473a781","chunk_id":"3830313A3231303864",
"quarantine_dir":"2F686F6D652F757365722F2E636F6D2E64727765622E71756172616E74696E65",
"restore_path":"2E2E2F7473742F65696361722E636F6D2E747874","creation_time":1462888884,
"report":{"object":"/home/user/tst/eicar.com.txt","size":68,"packer":[],
"virus":[{"type":"SE_KNOWN_VIRUS","name":"EICAR Test File (NOT a Virus!)"}]},
"heuristic_analysis":true,"core_fingerprint":"467CD4C6D423C55448B71CD5B8152776",
"item":[],"log":[],"user_time":0,"system_time":0,"stat":
{"dev":2049,"ino":898,
"size":68,"uid":{"uid":1000,"username":"user"},"gid":
{"gid":1000,"groupname":"user"},
"mode":33204,"mtime":1441028214,"ctime":1462888421,"xattr":[],"history":
[],
"detection_time":1462888667,"origin":"APP_COMMAND_LINE_TOOL"}
```

5. Change a configuration setting: disable Dr.Web CloudD.

A [LexMap](#) object in the JSON format must be written in advance to a file named `lexmap_ls_off.json`:

```
{"option":[{"key":"Root","map":{"option":
[{"key":"UseCloud","value":{"item":["no"]}}]}]}
```

Request:



```
$ curl -v -k -c cookie.jar -b cookie.jar -X POST -H 'Content-Type:
application/json' --data-binary '@lexmap_ls_off.json'
https://127.0.0.1:4443/api/10.2/set_lexmap
```

Response:

```
HTTP/1.0 200 OK
Content-Type: application/json
Content-Length: 58
Set-Cookie: DWToken=<...>; Secure; HttpOnly; Max-Age: 900; Path=/
Pragma: no-cache

{"item":[{"option":"Root.UseCloud","result":"EC_OK"}]}
```

6. Change a configuration setting: enable Dr.Web CloudD.

A [LexMap](#) object in the JSON format must be saved in a file named `lexmap_ls_on.json`:

```
{"option":[{"key":"Root","map":{"option":
[{"key":"UseCloud","value":{"item":["yes"]}}]}]}
```

Request:

```
$ curl -v -k -c cookie.jar -b cookie.jar -X POST -H 'Content-Type:
application/json' --data-binary '@lexmap_ls_on.json'
https://127.0.0.1:4443/api/10.2/set_lexmap
```

Response:

```
HTTP/1.0 200 OK
Content-Type: application/json
Content-Length: 58
Set-Cookie: DWToken=<...>; Secure; HttpOnly; Max-Age: 900; Path=/
Pragma: no-cache

{"item":[{"option":"Root.UseCloud","result":"EC_OK"}]}
```



Dr.Web SNMPD

Dr.Web SNMPD is an SNMP agent designed to integrate Dr.Web for UNIX Internet Gateways with monitoring systems running the SNMP protocol. This integration allows you to track the status of the Dr.Web for UNIX Internet Gateways components, as well as collect statistics on the detection and neutralization of threats. The agent supports providing monitoring systems or any SNMP managers with the following information:

- Status of any Dr.Web for UNIX Internet Gateways 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 Dr.Web for UNIX Internet Gateways components.
2. Dr.Web for UNIX Internet Gateways operation statistics.
3. Dr.Web for UNIX Internet Gateways component status.

For more details about information that can be obtained over the SNMP protocol, refer to the corresponding [section](#).

Operating Principles

In this section

- [General Information](#)
- [Integration with the System SNMP Agent](#)

General Information

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 Dr.Web for UNIX Internet Gateways components and notifications on detected threats directly from the [Dr.Web ConfigD](#) configuration daemon.

Threats can be detected by the scan engine during scanning initiated by Dr.Web for UNIX Internet Gateways components. Once any of threats is detected, 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 zero.

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 GNU/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 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
<code>--help</code>	Function: 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: 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 can be found in the [SNMPD] section of the integrated [configuration file](#) of Dr.Web for UNIX Internet Gateways.

The section contains the following parameters:

Parameter	Description
LogLevel <i>{logging level}</i>	Logging level of the component. If the parameter value is not specified, the <code>DefaultLogLevel</code> parameter value from the [Root] section is used. Default value: <code>Notice</code>
Log <i>{log type}</i>	Logging method of the component. Default value: <code>Auto</code>
ExePath	Path to the executable file of the component.



Parameter	Description
<i>{path to file}</i>	<p>Default value: <code><opt_dir>/bin/drweb-snmppd</code>.</p> <ul style="list-style-type: none">• For GNU/Linux: <code>/opt/drweb.com/bin/drweb-snmppd</code>.• For FreeBSD: <code>/usr/local/libexec/drweb.com/bin/drweb-snmppd</code>
Start <i>{Boolean}</i>	<p>Launch/do not launch the component by the Dr.Web ConfigD configuration daemon.</p> <p>When you specify the <code>Yes</code> value for this parameter, it the configuration daemon will start the component immediately; and when you specify the <code>No</code> value, the configuration daemon will terminate the component immediately.</p> <p>Default value: <code>No</code></p>
RunAsUser <i>{UID user name}</i>	<p>The user on whose behalf 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: <code>RunAsUser = name:123456</code>.</p> <p>When a user name is not specified, the component operation terminates with an error after the startup.</p> <p>Default value: <code>drweb</code></p>
ListenAddress <i>{address}</i>	<p>Address (IP address and port) listened by Dr.Web SNMPD, which is waiting for client connections (SNMP managers).</p> <p>Note that interaction with <code>snmpd</code> requires a specified port, different from the standard port (161), and <code>snmpd</code> must be configured for proxying.</p> <p>Default value: <code>127.0.0.1:161</code></p>
SnmVersion <i>{V2c V3}</i>	<p>The current version of SNMP protocol (<i>SNMPv2c</i> or <i>SNMPv3</i>).</p> <p>Default value: <code>V2c</code></p>
V3EngineId <i>{string}</i>	<p>Identifier (string) of <i>Engine ID</i> for <i>SNMPv3</i> (according to RFC 3411).</p> <p>Default value: <code>800073FA044452574542</code></p>
TrapReceiver <i>{address list}</i>	<p>List of addresses (IP address and port) where Dr.Web SNMPD sends <i>SNMP trap</i> notifications after Dr.Web for UNIX Internet Gateways components detected a threat.</p> <p>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</p>



Parameter	Description
	<p>than once in the section (in this case, all its values are combined into one list).</p> <p>Example: Add sockets 192.168.0.1:1234 and 10.20.30.45:5678 to the list.</p> <ol style="list-style-type: none">1. Adding of values to the configuration file.<ul style="list-style-type: none">• Two values in one string:<pre data-bbox="783 539 1449 678">Section [SNMPD] TrapReceiver = "192.168.0.1:1234", "10.20.30.45:5678"</pre>• Two strings (one value per a string):<pre data-bbox="783 734 1449 873">[SNMPD] TrapReceiver = 192.168.0.1:1234 TrapReceiver = 10.20.30.45:5678</pre>2. Adding values with the <code>drweb-ctl cfset</code> command:<pre data-bbox="783 936 1449 1104"># drweb-ctl cfset SNMPD.TrapReceiver -a 192.168.0.1:1234 # drweb-ctl cfset SNMPD.TrapReceiver -a 10.20.30.45:5678</pre> <p>Default value: <i>(not set)</i></p>
V2cCommunity <i>{string}</i>	<p>The string “SNMP read community” for authentication of SNMP managers (<i>SNMPv2c</i> protocol) when Dr.Web MIB variables are accessed for reading.</p> <p>The parameter is used if <code>SnmVersion = V2c</code>.</p> <p>Default value: <code>public</code></p>
V3UserName <i>{string}</i>	<p>The user name for authentication of SNMP managers (<i>SNMPv3</i> protocol) when Dr.Web MIB variables are accessed for reading.</p> <p>The parameter is used if <code>SnmVersion = V3</code>.</p> <p>Default value: <code>noAuthUser</code></p>
V3Auth <i>{SHA(<pwd>) MD5(<pwd>) None}</i>	<p>Method to authenticate SNMP managers (<i>SNMPv3</i> protocol) when Dr.Web MIB variables are accessed for reading.</p> <p>Allowed values:</p> <ul style="list-style-type: none">• <code>SHA (<PWD>)</code>—SHA hash of the password is used (<PWD> strings);• <code>MD5 (<PWD>)</code>—MD5 hash of the password is used (<PWD> strings);



Parameter	Description
	<ul style="list-style-type: none">• <code>None</code>—authentication is disabled. <p>where <code><PWD></code> is a plain text password.</p> <p>When specifying the parameter value from the command line, you may need to escape the brackets by using the slash mark <code>\</code> in some shells.</p> <p>Examples:</p> <ol style="list-style-type: none">1. Parameter value in the configuration file: <code>V3Auth = MD5(123456)</code>2. Specifying the same parameter value from the command line with the <code>drweb-ctl cfset</code> command: <code>drweb-ctl cfset SNMPD.V3Auth MD5\ (123456\)</code> <p>The parameter is used if <code>SnmpVersion = V3</code>.</p> <p>Default value: <code>None</code></p>
<code>V3Privacy</code> <code>{DES(<secret>) AES128(<secret>) None}</code>	<p>Encryption method for SNMP messages (<i>SNMPv3</i> protocol).</p> <p>Allowed values:</p> <ul style="list-style-type: none">• <code>DES (<secret>)</code>—DES encryption algorithm;• <code>AES128 (<secret>)</code>—AES128 encryption algorithm;• <code>None</code>—SNMP-messages are not encrypted. <p>where <code><secret></code> is a secret key shared by the manager and the agent (<i>plain text</i>).</p> <p>When specifying the parameter value from the command line, you can need to escape the brackets by using the slash mark <code>\</code> in some shells.</p> <p>Examples:</p> <ol style="list-style-type: none">1. Parameter value in the configuration file: <code>V3Privacy = AES128(supersecret)</code>2. Specifying the same parameter value from the command line with the <code>drweb-ctl cfset</code> command: <code>drweb-ctl cfset SNMPD.V3Privacy AES128\ (supersecret\)</code> <p>The parameter is used if <code>SnmpVersion = V3</code>.</p> <p>Default value: <code>None</code></p>



Integration with SNMP Monitoring Systems

Dr.Web SNMP agent can act as data provider for any monitoring system that uses the SNMP protocol, versions 2c or 3. The list of data that is available for control and the data structure are [described](#) in the Dr.Web MIB description file `DRWEB-SNMPD-MIB.txt` supplied together with Dr.Web for UNIX Internet Gateways. This file is located in the directory `<opt_dir>/share/drweb-snmpd/mibs`.

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* (*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 (it is possible to get email statistics only with the Dr.Web MailD component. Dr.Web MailD is not included in Dr.Web for UNIX Internet Gateways).

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
<code>plugins/snmp__drweb_malware</code>	The <code>munin-node</code> plug-in for polling Dr.Web SNMPD via SNMP to gather the number of threats detected



File	Description
	by Dr.Web for UNIX Internet Gateways on the host.
<code>plugins/snmp__drweb_filecheck</code>	The munin-node plug-in for polling Dr.Web SNMPD via SNMP to gather the statistics of files scanned by Dr.Web for UNIX Internet Gateways on the host.
<code>plugin-conf.d/drweb.cfg</code>	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, and so on. 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 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://guide.munin-monitoring.org/en/latest>.

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-snmptt/connectors/zabbix` directory.

File	Description
<code>zbx_drweb.xml</code>	Template for description of the monitored host that features installed Dr.Web for UNIX Internet Gateways
<code>snmptt.drweb.zabbix.conf</code>	Configuring the <code>snmptt</code> utility—which is an <i>SNMP</i> 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-snmppd/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-snmppd` 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:

Press **Add**, check *DRWEB*, press **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**.
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 versions 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 versions 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.

In **Severity**, for this trigger it is recommended that notification type is different from *Not classified*, 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, `snmpthandler`, `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 (the `item drweb-traps` element). The *SNMP trap* format of the saved notification is specified in the `<opt_dir>/share/drweb-`



snmpd/connectors/zabbix/snmppt.drweb.zabbix.conf. file. The file must be copied to /etc/snmp.

- Moreover, the path to the format files must be specified in the snmppt.ini:

```
[TrapFiles]
# A list of snmppt.conf files (this is NOT the snmptrapd.conf file).
# The COMPLETE path and filename. Ex: '/etc/snmp/snmppt.conf'
snmppt_conf_files = <<END
/etc/snmp/snmppt.conf
/etc/snmp/snmppt.drweb.zabbix.conf
END
```

After that, restart snmppt if it was started in daemon mode.

- 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/snmppt/snmppt.log
StartSNMPTrapper=1
```

where /var/log/snmppt/snmppt.log is a log file used by snmppt to register information on received SNMP trap notifications.

For official documentation on Zabbix, refer to <https://www.zabbix.com/documentation/current/en>.

Integration with Nagios Monitoring System

Files with Nagios 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 <i>drweb</i> 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 for UNIX Internet Gateways is installed
plugins/eventhandlers/submit_check_result	The script for handling <i>SNMP trap</i> notifications
snmp/snmppt.drweb.nagios.conf	Configuring the snmppt utility—which is an <i>SNMP trap</i> 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 Nagiosplug-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 `perfllog` 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 for UNIX Internet Gateways 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 `snmpwalk` 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_perfddata_file=/usr/nagiosgraph/var/rrd/perfddata.log
service_perfddata_file_template=$LASTSERVICECHECK$||$HOSTNAME$||$SERVICEDES
C$||$SERVICEOUTPUT$||$SERVICEPERFDATA$
service_perfddata_file_mode=a
service_perfddata_file_processing_interval=30
service_perfddata_file_processing_command=process-service-perfddata-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-snmppd/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/snmpptt.drweb.nagios.conf > /etc/snmp/snmp/snmpptt.drweb.nagios.conf
```

4. Add the `" /etc/snmp/snmpptt.drweb.nagios.conf"` string to the `/etc/snmp/snmpptt.ini` 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 <https://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: <code>.iso.org.dod.internet.private.enterprises.drweb.drwebSnmpd</code> Common prefix for all OIDs: <code>.1.3.6.1.4.1.29690.2</code>		
alert	Asynchronous notifications about events (SNMP traps)	
<code>threatAlert</code>	<code>.1.1</code>	Notification about detecting a threat
<code>threatAlertFile</code>	<code>.1.1.1</code>	Name of the infected file (string)
<code>threatAlertType</code>	<code>.1.1.2</code>	Threat type (integer *)



Parameter name	OID of the parameter	Type and description of the parameter
<i>threatAlertName</i>	.1.1.3	Name of the threat (string)
<i>threatAlertOrigin</i>	.1.1.4	Identifier of the component that detected the threat (integer***)
<i>threatAlertRemotelp</i>	.1.1.5	IP address of the remote host that was used to access the file (string)
<i>threatAlertRemoteUser</i>	.1.1.6	Name of the remote user that accessed the file (string)
<i>threatAlertRemoteDomain</i>	.1.1.7	Name of the remote host that was used to access the file (string)
threatActionErrorAlert	.1.2	Notification about an error occurred when trying to neutralize the threat
<i>threatActionErrorAlertFile</i>	.1.2.1	Name of the infected file (string)
<i>threatActionErrorAlertType</i>	.1.2.2	Threat type (integer *)
<i>threatActionErrorAlertName</i>	.1.2.3	Name of the threat (string)
<i>threatActionErrorAlertOrigin</i>	.1.2.4	Identifier of the component that detected the threat (integer***)
<i>threatActionErrorAlertError</i>	.1.2.5	Description of an error (string)
<i>threatActionErrorAlertErrorCode</i>	.1.2.6	Error code (integer corresponding to code from error catalogue)
<i>threatActionErrorAlertAction</i>	.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
<i>componentFailureAlertName</i>	.1.3.1	Component identifier (integer***)
<i>componentFailureAlertExitCodeDescription</i>	.1.3.2	Component exit code description (string)
<i>componentFailureAlertExitCode</i>	.1.3.3	Error code (integer corresponding to code from error catalogue)



Parameter name	OID of the parameter	Type and description of the parameter
<code>infectedUrlAlert</code>	.1.4	Notification about blocking a malicious URL (for HTTP/HTTPS connections)
<code>infectedUrlAlertUrl</code>	.1.4.1	The URL that was blocked (string)
<code>infectedUrlAlertDirection</code>	.1.4.2	HTTP message direction (integer: 1—request, 2—response)
<code>infectedUrlAlertType</code>	.1.4.3	Threat type (integer *)
<code>infectedUrlAlertName</code>	.1.4.4	Name of the threat (string)
<code>infectedUrlAlertOrigin</code>	.1.4.5	Identifier of the component that detected the threat (integer***)
<code>infectedUrlAlertSrcIp</code>	.1.4.6	IP address of connection source (string)
<code>infectedUrlAlertSrcPort</code>	.1.4.7	Port of connection source (integer)
<code>infectedUrlAlertDstIp</code>	.1.4.8	IP address of connection destination point (string)
<code>infectedUrlAlertDstPort</code>	.1.4.9	Port of connection destination point (integer)
<code>infectedUrlAlertSniHost</code>	.1.4.10	SNI of connection destination point (for SSL connections) (string)
<code>infectedUrlAlertExePath</code>	.1.4.11	Executable path of the program that establish the connection (string)
<code>infectedUrlAlertUserName</code>	.1.4.12	Name of the user with whose privileges is executing the program that establish the connection (string)
<code>infectedEmailAttachmentAlert</code>	.1.5	Notification about detecting an infected email attachment
<code>infectedEmailAttachmentAlertType</code>	.1.5.1	Threat type (integer *)
<code>infectedEmailAttachmentAlertName</code>	.1.5.2	Name of the threat (string)
<code>infectedEmailAttachmentAlertOrigin</code>	.1.5.3	Identifier of the component that detected the threat (integer***)



Parameter name	OID of the parameter	Type and description of the parameter
<i>infectedEmailAttachmentAlertSocket</i>	.1.5.4	IP address of the source of the email message (string)
<i>infectedEmailAttachmentAlertMailFrom</i>	.1.5.5	Sender of the email message (string)
<i>infectedEmailAttachmentAlertRcptTo</i>	.1.5.6	Recipients of the email message (string)
<i>infectedEmailAttachmentAlertMessageId</i>	.1.5.7	Value of <code>Message-ID</code> header of the email message (string)
<i>infectedEmailAttachmentAlertAction</i>	.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)
<i>infectedEmailAttachmentAlertDivert</i>	.1.5.9	Direction of the email message (integer: 1—incoming; 2—outgoing)
<i>infectedEmailAttachmentAlertSrcIp</i>	.1.5.10	IP address of connection source (string)
<i>infectedEmailAttachmentAlertSrcPort</i>	.1.5.11	Port of connection source (integer)
<i>infectedEmailAttachmentAlertDstIp</i>	.1.5.12	IP address of connection destination point (string)
<i>infectedEmailAttachmentAlertDstPort</i>	.1.5.13	Port of connection destination point (integer)
<i>infectedEmailAttachmentAlertSniHost</i>	.1.5.14	SNI of connection destination point (for SSL connections) (string)
<i>infectedEmailAttachmentAlertProtocol</i>	.1.5.15	Protocol type (integer: 1—SMTP; 2—POP3; 3—IMAP; 4—HTTP)
<i>infectedEmailAttachmentAlertExePath</i>	.1.5.16	Executable path of the program that establish the connection (string)
<i>infectedEmailAttachmentAlertUserName</i>	.1.5.17	Name of the user with whose privileges is executing the program that establish the connection (string)



Parameter name	OID of the parameter	Type and description of the parameter
categoryUrlAlert	.1.6	Notification about blocking a URL belonging to the unwanted category
<i>categoryUrlAlertUrl</i>	.1.6.1	The URL that was blocked (string)
<i>categoryUrlAlertCategory</i>	.1.6.2	Web resource category to which the URL belongs (integer**)
<i>categoryUrlAlertOrigin</i>	.1.6.3	Identifier of the component that detected the threat (integer***)
<i>categoryUrlAlertSrcIp</i>	.1.6.4	IP address of connection source (string)
<i>categoryUrlAlertSrcPort</i>	.1.6.5	Port of connection source (integer)
<i>categoryUrlAlertDstIp</i>	.1.6.6	IP address of connection destination point (string)
<i>categoryUrlAlertDstPort</i>	.1.6.7	Port of connection destination point (integer)
<i>categoryUrlAlertSniHost</i>	.1.6.8	SNI of connection destination point (for SSL connections) (string)
<i>categoryUrlAlertExePath</i>	.1.6.9	Executable path of the program that establish the connection (string)
<i>categoryUrlAlertUserName</i>	.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
<i>categoryUrlEmailAttachmentAlertType</i>	.1.7.1	Web resource category to which the URL belongs (integer**)
<i>categoryUrlEmailAttachmentAlertOrigin</i>	.1.7.2	Identifier of the component that detected the threat (integer***)
<i>categoryUrlEmailAttachmentAlertSocket</i>	.1.7.3	IP address of the source of the email message (string)



Parameter name	OID of the parameter	Type and description of the parameter
<i>categoryUrlEmailAttachmentAlertMailFrom</i>	.1.7.4	Sender of the email message (string)
<i>categoryUrlEmailAttachmentAlertRcptTo</i>	.1.7.5	Recipients of the email message (string)
<i>categoryUrlEmailAttachmentAlertMessageId</i>	.1.7.6	Value of <code>Message-ID</code> header of the email message (string)
<i>categoryUrlEmailAttachmentAlertAction</i>	.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)
<i>categoryUrlEmailAttachmentAlertDivert</i>	.1.7.8	Direction of the email message (integer: 1—incoming; 2—outgoing)
<i>categoryUrlEmailAttachmentAlertSrcIp</i>	.1.7.9	IP address of connection source (string)
<i>categoryUrlEmailAttachmentAlertSrcPort</i>	.1.7.10	Port of connection source (integer)
<i>categoryUrlEmailAttachmentAlertDstIp</i>	.1.7.11	IP address of connection destination point (string)
<i>categoryUrlEmailAttachmentAlertDstPort</i>	.1.7.12	Port of connection destination point (integer)
<i>categoryUrlEmailAttachmentAlertSniHost</i>	.1.7.13	SNI of connection destination point (for SSL connections) (string)
<i>categoryUrlEmailAttachmentAlertProtocol</i>	.1.7.14	Protocol type (integer: 1—SMTP; 2—POP3; 3—IMAP; 4—HTTP)
<i>categoryUrlEmailAttachmentAlertExePath</i>	.1.7.15	Executable path of the program that establish the connection (string)
<i>categoryUrlEmailAttachmentAlertUserName</i>	.1.7.16	Name of the user with whose privileges is executing the program that establish the connection (string)
<i>spamEmailAlert</i>	.1.8	Notification about recognizing an email message as spam



Parameter name	OID of the parameter	Type and description of the parameter
<i>spamEmailAlertOrigin</i>	.1.8.1	Identifier of the component that detected the threat (integer ^{***})
<i>spamEmailAlertSocket</i>	.1.8.2	IP address of the source of the email message (string)
<i>spamEmailAlertMailFrom</i>	.1.8.3	Sender of the email message (string)
<i>spamEmailAlertRcptTo</i>	.1.8.4	Recipients of the email message (string)
<i>spamEmailAlertMessageld</i>	.1.8.5	Value of <code>Message-ID</code> header of the email message (string)
<i>spamEmailAlertAction</i>	.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)
<i>spamEmailAlertDivert</i>	.1.8.7	Direction of the email message (integer: 1—incoming; 2—outgoing)
<i>spamEmailAlertSrcIp</i>	.1.8.8	IP address of connection source (string)
<i>spamEmailAlertSrcPort</i>	.1.8.9	Port of connection source (integer)
<i>spamEmailAlertDstIp</i>	.1.8.10	IP address of connection destination point (string)
<i>spamEmailAlertDstPort</i>	.1.8.11	Port of connection destination point (integer)
<i>spamEmailAlertSniHost</i>	.1.8.12	SNI of connection destination point (for SSL connections) (string)
<i>spamEmailAlertProtocol</i>	.1.8.13	Protocol type (integer: 1—SMTP; 2—POP3; 3—IMAP; 4—HTTP)
<i>spamEmailAlertExePath</i>	.1.8.14	Executable path of the program that establish the connection (string)



Parameter name	OID of the parameter	Type and description of the parameter
<i>spamEmailAlertUserName</i>	.1.8.15	Name of the user with whose privileges is executing the program that establish the connection (string)
<i>blockedConnectionAlert</i>	.1.9	Notification about blocking a network connection
<i>blockedConnectionAlertOrigin</i>	.1.9.1	Identifier of the component that detected the threat (integer ^{***})
<i>blockedConnectionAlertDivert</i>	.1.9.2	Direction of the connection (integer: 1—incoming; 2—outgoing)
<i>blockedConnectionAlertSrcIp</i>	.1.9.3	IP address of connection source (string)
<i>blockedConnectionAlertSrcPort</i>	.1.9.4	Port of connection source (integer)
<i>blockedConnectionAlertDstIp</i>	.1.9.5	IP address of connection destination point (string)
<i>blockedConnectionAlertDstPort</i>	.1.9.6	Port of connection destination point (integer)
<i>blockedConnectionAlertSniHost</i>	.1.9.7	SNI of connection destination point (for SSL connections) (string)
<i>blockedConnectionAlertProtocol</i>	.1.9.8	Protocol type (integer: 1—SMTP; 2—POP3; 3—IMAP; 4—HTTP)
<i>blockedConnectionAlertExePath</i>	.1.9.9	Executable path of the program that establish the connection (string)
<i>blockedConnectionAlertUserName</i>	.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 Dr.Web for UNIX Internet Gateways	
<i>threatCounters</i>	.2.1	Counters of detected threats
<i>knownVirus</i>	.2.1.1	Number of detected known viruses (counter; integer)



Parameter name	OID of the parameter	Type and description of the parameter
<i>suspicious</i>	.2.1.2	Number of detected suspicious objects (counter; integer)
<i>adware</i>	.2.1.3	Number of detected adware (counter; integer)
<i>dialers</i>	.2.1.4	Number of detected dialers (counter; integer)
<i>joke</i>	.2.1.5	Number of detected joke programs (counter; integer)
<i>riskware</i>	.2.1.6	Number of detected riskware (counter; integer)
<i>hacktool</i>	.2.1.7	Number of detected hacktools (counter; integer)
scanErrors	.2.2	Counters of the errors that occurred while files were scanned
<i>sePathNotAbsolute</i>	.2.2.1	Number of occurrences of the "Path is not absolute" error (counter, integer)
<i>seFileNotFound</i>	.2.2.2	Number of occurrences of the "File not found" error (counter, integer)
<i>seFileNotRegular</i>	.2.2.3	Number of occurrences of the "File is not a regular file" error (counter, integer)
<i>seFileNotBlockDevice</i>	.2.2.4	Number of occurrences of the "File is not a block device" error (counter, integer)
<i>seNameTooLong</i>	.2.2.5	Number of occurrences of the "Path or file name is too long" error (counter, integer)
<i>seNoAccess</i>	.2.2.6	Number of occurrences of the "Permission denied" error (counter, integer)
<i>seReadError</i>	.2.2.7	Number of occurrences of the "Read error" (counter, integer)
<i>seWriteError</i>	.2.2.8	Number of occurrences of the "Write error" (counter, integer)



Parameter name	OID of the parameter	Type and description of the parameter
<i>seFileTooLarge</i>	.2.2.9	Number of occurrences of the "File size too big" error (counter, integer)
<i>seFileBusy</i>	.2.2.10	Number of occurrences of the "File is busy" error (counter, integer)
<i>seUnpackingError</i>	.2.2.20	Number of occurrences of the "Unpacking error" (counter, integer)
<i>sePasswordProtectd</i>	.2.2.21	Number of occurrences of the "Password protected" error (counter, integer)
<i>seArchCrcError</i>	.2.2.22	Number of occurrences of the "Archive Cyclic Redundancy Check error" (counter, integer)
<i>seArchInvalidHeader</i>	.2.2.23	Number of occurrences of the "Invalid archive header" error (counter, integer)
<i>seArchNoMemory</i>	.2.2.24	Number of occurrences of the "Not enough memory to process the archive" error (counter, integer)
<i>seArchIncomplete</i>	.2.2.25	Number of occurrences of the "Incomplete archive" error (counter, integer)
<i>seCanNotBeCured</i>	.2.2.26	Number of occurrences of the "Object cannot be cured" error (counter, integer)
<i>sePackerLevelLimit</i>	.2.2.30	Number of occurrences of the error that states that the maximum nesting level of packed objects was exceeded (counter, integer)
<i>seArchiveLevelLimit</i>	.2.2.31	Number of occurrences of the error that states that the maximum nesting level of archives was exceeded (counter, integer)
<i>seMailLevelLimit</i>	.2.2.32	Number of occurrences of the error that states that the



Parameter name	OID of the parameter	Type and description of the parameter
		maximum nesting level of email files was exceeded (counter, integer)
<i>seContainerLevelLimit</i>	.2.2.33	Number of occurrences of the error that states that the maximum nesting level of container files was exceeded (counter, integer)
<i>seCompressionLimit</i>	.2.2.34	Number of occurrences of the "Exceeded the maximum compression ratio" error (counter, integer)
<i>seReportSizeLimit</i>	.2.2.35	Number of occurrences of the "Exceeded the maximum size of the scanning results report" error (counter, integer)
<i>seScanTimeout</i>	.2.2.40	Number of occurrences of the "Scan timeout expired" error (counter, integer)
<i>seEngineCrash</i>	.2.2.41	Number of occurrences of the "Scan Engine crash was detected" error (counter, integer)
<i>seEngineHangup</i>	.2.2.42	Number of occurrences of the "Scan Engine stopped responding" error (counter, integer)
<i>seEngineError</i>	.2.2.44	Number of occurrences of the "Internal error of the Scan Engine" (counter, integer)
<i>seNoLicense</i>	.2.2.45	Number of occurrences of the "No valid license found" error (counter, integer)
<i>seCuringLimitReached</i>	.2.2.47	Number of occurrences of the "Curing attempts limit is reached" error (counter, integer)
<i>seNonSupportedDisk</i>	.2.2.50	Number of Occurrences of the "Unsupported disk" error (counter, integer)



Parameter name	OID of the parameter	Type and description of the parameter
<i>seUnexpectedError</i>	.2.2.100	Number of occurrences of the "Unexpected error" (counter, integer)
<i>scanLoadAverage</i>	.2.3	Metrics of the file scanning load
<i>filesScannedTable</i>	.2.3.1	Average numbers of files scanned at the request of other components
<i>filesScannedEntry</i>	.2.3.1.1	Component of the product (entire table row, record)
<i>filesScannedIndex</i>	.2.3.1.1.1	Index of the component (identifier, integer***)
<i>filesScannedOrigin</i>	.2.3.1.1.2	Name of the component
<i>filesScanned1min</i>	.2.3.1.1.3	The average (averaged over one minute) number of files checked per second (string)
<i>filesScanned5min</i>	.2.3.1.1.4	The average (averaged over 5 minutes) number of files checked per second (string)
<i>filesScanned15min</i>	.2.3.1.1.5	The average (averaged over 15 minutes) number of files checked per second (string)
<i>bytesScannedTable</i>	.2.3.2	Average speed (in bytes per second) of scanning performed at the request of other components
<i>bytesScannedEntry</i>	.2.3.2.1	Component of the product (entire table row, record)
<i>bytesScannedIndex</i>	.2.3.2.1.1	Index of the component (identifier, integer***)
<i>bytesScannedOrigin</i>	.2.3.2.1.2	Name of the component
<i>bytesScanned1min</i>	.2.3.2.1.3	The average (averaged over one minute) number of bytes scanned per second (string)
<i>bytesScanned5min</i>	.2.3.2.1.4	The average (averaged over 5 minutes) number of bytes scanned per second (string)



Parameter name	OID of the parameter	Type and description of the parameter
bytesScanned15min	.2.3.2.1.5	The average (averaged over 15 minutes) number of bytes scanned per second (string)
<i>cacheHitFilesTable</i>	.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
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)
<i>errorsTable</i>	.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



Parameter name	OID of the parameter	Type and description of the parameter
		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
<i>markedAsSpam</i>	.2.4.1	Number of email messages marked as spam (counter, integer)
<i>blockedInfectionSource</i>	.2.4.101	Number of blocked URLs belonging to the "Infection Source" category (counter, integer)
<i>blockedNotRecommended</i>	.2.4.102	Number of blocked URLs belonging to the "Not Recommended" category (counter, integer)
<i>blockedAdultContent</i>	.2.4.103	Number of blocked URLs belonging to the "Adult Content" category (counter, integer)
<i>blockedViolence</i>	.2.4.104	Number of blocked URLs belonging to the "Violence" category (counter, integer)
<i>blockedWeapons</i>	.2.4.105	Number of blocked URLs belonging to the "Weapons" category (counter, integer)
<i>blockedGambling</i>	.2.4.106	Number of blocked URLs belonging to the "Gambling" category (counter, integer)
<i>blockedDrugs</i>	.2.4.107	Number of blocked URLs belonging to the "Drugs" category (counter, integer)
<i>blockedObsceneLanguage</i>	.2.4.108	Number of blocked URLs belonging to the "Obscene Language" category (counter, integer)
<i>blockedChats</i>	.2.4.109	Number of blocked URLs belonging to the "Chats" category (counter, integer)



Parameter name	OID of the parameter	Type and description of the parameter
<i>blockedTerrorism</i>	.2.4.110	Number of blocked URLs belonging to the "Terrorism" category (counter, integer)
<i>blockedFreeEmail</i>	.2.4.111	Number of blocked URLs belonging to the "Free Email Services" category (counter, integer)
<i>blockedSocialNetworks</i>	.2.4.112	Number of blocked URLs belonging to the "Social Networks" category (counter, integer)
<i>blockedOwnersNotice</i>	.2.4.113	Number of blocked URLs belonging to the "Copyright Owner's Notice" category (counter, integer)
<i>blockedOnlineGames</i>	.2.4.114	Number of blocked URLs belonging to the "Online games" category (counter, integer)
<i>blockedAnonymizers</i>	.2.4.115	Number of blocked URLs belonging to the "Anonymizers" category (counter, integer)
<i>blockedCryptocurrencyMiningPools</i>	.2.4.116	Number of blocked URLs belonging to the "Cryptocurrency mining pools" category (counter, integer)
<i>blockedJobs</i>	.2.4.117	Number of blocked URLs belonging to the "Job search sites" category (counter, integer)
<i>blockedBlackList</i>	.2.4.120	Number of blocked URLs from the user's black list (counter, integer)
info	Information about the current state of Dr.Web for UNIX Internet Gateways	
components	.3.1	Dr.Web for UNIX Internet Gateways component status
<i>configd</i>	.3.1.1	drweb-configd component data



Parameter name	OID of the parameter	Type and description of the parameter
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 (<i>UNIX time</i>)
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	Executable path to the component (string)
configdAppInstallTime	.3.1.1.101.1.4	Time when the component was installed (<i>UNIX time</i>)
configdAppIniSection	.3.1.1.101.1.5	Name of the section with the component 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 (<i>UNIX time</i>)
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)



Parameter name	OID of the parameter	Type and description of the parameter
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)
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)



Parameter name	OID of the parameter	Type and description of the parameter
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 (<i>UNIX time</i>)
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)
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)
<i>fileCheck</i>	.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 (<i>UNIX time</i>)



Parameter name	OID of the parameter	Type and description of the parameter
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 scan 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)
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 scan engine when working for this client (counter, integer)
<i>update</i>	.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 (<i>UNIX time</i>)



Parameter name	OID of the parameter	Type and description of the parameter
updateBytesSent	.3.1.4.101	Number of bytes sent (counter, integer)
updateBytesReceived	.3.1.4.102	Number of bytes received (counter, integer)
<i>esagent</i>	.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 (<i>UNIX time</i>)
esagentWorkStatus	.3.1.5.101	The component current mode of operation (integer: 1—standalone mode, 2—is connecting, 3—is awaiting connection, 4—connection has been approved)
esagentIsConnected	.3.1.5.102	Connected to the server (integer, 0—no, 1—yes)
esagentServer	.3.1.5.103	Address of the centralized protection server that is used (string)
<i>netcheck</i>	.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 (<i>UNIX time</i>)
netcheckLocalSeForks	.3.1.6.101	The number of scan engine processes available locally (integer)



Parameter name	OID of the parameter	Type and description of the parameter
netcheckRemoteSeForks	.3.1.6.102	Number of scan 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)
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 scan engine processes (counter, integer)
netcheckNetworkScanErrors	.3.1.6.112	Number of error occurrences in remote scan engine processes (counter, integer)
<i>httpd</i>	.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)



Parameter name	OID of the parameter	Type and description of the parameter
httpdExitTime	.3.1.7.3	Time of the last termination (<i>UNIX time</i>)
<i>snmpd</i>	.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 (<i>UNIX time</i>)
<i>clamd</i>	.3.1.20	drweb-clamd component data
clamdState	.3.1.20.1	Current state of the component (integer****)
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 (<i>UNIX time</i>)
<i>icapd</i>	.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 (<i>UNIX time</i>)
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 <i>OPTIONS</i> requests (counter, integer)



Parameter name	OID of the parameter	Type and description of the parameter
icapdReqmod	.3.1.21.104	Number of <i>REQMOD</i> requests (counter, integer)
icapdRespmod	.3.1.21.105	Number of <i>RESPMOD</i> requests (counter, integer)
icapdBad	.3.1.21.106	Number of invalid requests (counter, integer)
<i>smbspider</i>	.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)
smbspiderExitTime	.3.1.40.3	Time of the last termination (<i>UNIX time</i>)
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)
<i>gated</i>	.3.1.41	drweb-gated component data



Parameter name	OID of the parameter	Type and description of the parameter
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 (<i>UNIX time</i>)
gatedInterceptedIn	.3.1.41.101	Number of intercepted connections (counter, integer)
gatedInterceptedCount	.3.1.41.102	Number of currently monitored connections (counter, integer)
<i>maild</i>	.3.1.42	drweb-maild component data
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 (<i>UNIX time</i>)
maildStat	.3.1.42.4	Statistics of the drweb-maild component operation
maildStatNative	.3.1.42.4.1	Email scanning statistics via the component internal interface drweb-maild (messages received by SplDer Gate during the scan of intercepted 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)



Parameter name	OID of the parameter	Type and description of the parameter
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 interface <i>Milter</i> of the <i>drweb-maild</i> 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)
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 interface <i>Spamc</i> of the <i>drweb-maild</i> 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 interface <i>Rspamc</i> of the <i>drweb-maild</i> component



Parameter name	OID of the parameter	Type and description of the parameter
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)
<i>lookupd</i>	.3.1.43	drweb-lookupd component data
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 (<i>UNIX time</i>)
<i>antispam</i>	.3.1.44	Data about the drweb-ase component
antispamState	.3.1.44.1	Current state of the component (integer****)
antispamExitCode	.3.1.44.2	Last exit code (integer corresponding to code from error catalogue)
antispamExitTime	.3.1.44.3	Time of the last termination (<i>UNIX time</i>)
<i>cloudd</i>	.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



Parameter name	OID of the parameter	Type and description of the parameter
		error catalogue)
clouddExitTime	.3.1.50.3	Time of the last termination (<i>UNIX time</i>)
<i>meshd</i>	.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 (<i>UNIX time</i>)
<i>lotus</i>	.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 (<i>UNIX time</i>)
<i>macgui</i>	.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 (<i>UNIX time</i>)
<i>macspider</i>	.3.1.102	drweb-spider (for macOS) component data
macspiderState	.3.1.102.1	Current state of the component (integer****)



Parameter name	OID of the parameter	Type and description of the parameter
macspiderExitCode	.3.1.102.2	Last exit code (integer corresponding to code from error catalogue)
macspiderExitTime	.3.1.102.3	Time of the last termination (<i>UNIX time</i>)
macspiderWorkStatus	.3.1.102.101	The component current mode of operation (integer: 0—not set, 1—loading, 2—is running)
<i>macfirewall</i>	.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 (<i>UNIX time</i>)
<i>linuxgui</i>	.3.1.200	drweb-gui (for GNU/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 (<i>UNIX time</i>)
<i>linuxspider</i>	.3.1.201	drweb-spider (for GNU/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 (<i>UNIX time</i>)



Parameter name	OID of the parameter	Type and description of the parameter
linuxspiderWorkStatus	.3.1.201.101	The component current mode of operation (integer: 0—not set, 1—loading, 2—running via fanotify, 3—running via LKM)
<i>linuxnss</i>	.3.1.202	drweb-nss (for GNU/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 (<i>UNIX time</i>)
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)
<i>linuxfirewall</i>	.3.1.203	drweb-firewall (for GNU/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 (<i>UNIX time</i>)
<i>ctl</i>	.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 (<i>UNIX time</i>)
license	.3.2	License status
<i>licenseEsMode</i>	.3.2.1	The license has been granted by the centralized protection server (integer: 0—no, 1—yes)
<i>licenseNumber</i>	.3.2.2	License number (integer)
<i>licenseOwner</i>	.3.2.3	License owner (string)
<i>licenseActivated</i>	.3.2.4	License activation date (<i>UNIX time</i>)
<i>licenseExpires</i>	.3.2.5	License expiration date (<i>UNIX time</i>)

*) Threat types:

Code	Threat Type
1	Known virus (<i>known virus</i>)
2	Suspicious object (<i>suspicious</i>)
3	Adware (<i>adware</i>)
4	Dialer (<i>dialer</i>)
5	Joke program (<i>joke program</i>)
6	Riskware (<i>riskware</i>)
7	Hacktool (<i>hacktool</i>)

**) Categories of URL:

Code	Threat Type
1	Infection source (<i>infectionSource</i>)
2	Not recommended (<i>notRecommended</i>)
3	Adult content (<i>adultContent</i>)
4	Violence (<i>violence</i>)



Code	Threat Type
5	Weapons (<i>weapons</i>)
6	Gambling (<i>gambling</i>)
7	Drugs (<i>drugs</i>)
8	Obscene language (<i>obsceneLanguage</i>)
9	Chats (<i>chats</i>)
10	Terrorism (<i>terrorism</i>)
11	Free email (<i>freeEmail</i>)
12	Social networks (<i>socialNetworks</i>)
13	URL added due to a notice from copyright owner (<i>ownerNotice</i>)
14	Online games (<i>onlineGames</i>)
15	Anonymizers (<i>anonymizers</i>)
16	Cryptocurrency mining pools (<i>cryptocurrencyMiningPools</i>)
17	Job search sites (<i>Jobs</i>)
20	Added to black list (<i>blackList</i>)

***) Codes of Dr.Web components:

Code	Component
1	Dr.Web ConfigD (<i>drweb-configd</i>)
2	Dr.Web Scanning Engine (<i>drweb-se</i>)
3	Dr.Web File Checker (<i>drweb-filecheck</i>)
4	Dr.Web Updater (<i>drweb-update</i>)
5	Dr.Web ES Agent (<i>drweb-esagent</i>)
6	Dr.Web Network Checker (<i>drweb-netcheck</i>)
7	Dr.Web HTTPD (<i>drweb-httpd</i>)
8	Dr.Web SNMPD (<i>drweb-snmpd</i>)
20	Dr.Web ClamD (<i>drweb-clamd</i>)



Code	Component
21	Dr.Web ICAPD (drweb-icapd)
40	SplDer Guard for SMB (drweb-smbspider-daemon)
41	SplDer Gate (drweb-gated)
42	Dr.Web MailD (drweb-maild)
43	Dr.Web LookupD (drweb-lookupd)
50	Dr.Web CloudD (drweb-cloudd)
52	Dr.Web MeshD (drweb-meshd)
60	Dr.Web for Lotus
100	drweb-gui for macOS
102	SplDer Guard for macOS for macOS
103	Dr.Web Firewall for macOS for macOS
200	drweb-gui for GNU/Linux
201	SplDer Guard (drweb-spider)
202	SplDer Guard for NSS (drweb-nss)
203	Dr.Web Firewall for Linux (drweb-firewall) for GNU/Linux
300	Dr.Web Ctl (drweb-ctl)
400	Enterprise scanner (not a component of Dr.Web for UNIX Internet Gateways)

****) 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 host, 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 MeshD

Dr.Web MeshD is an agent that includes the host with Dr.Web for UNIX Internet Gateways installed in a “local cloud” that connects hosts with installed Dr.Web for UNIX products. This cloud allows to solve the following tasks:

- providing other file scanning services by several cloud hosts (a scanning core service);
- distributing updates for virus databases between cloud hosts.

To connect hosts with Dr.Web for UNIX products installed, the Dr.Web MeshD component must be installed on every host. The component includes the host into the cloud. Host’s rights within the cloud and cloud features that are used by the host, can be easily configured by Dr.Web MeshD settings.

Data is shared with other cloud hosts over a protected SSH channel.

Operating Principles

In this section

- [Connection types](#)
- [Operation modes](#)
- [Services](#)

Dr.Web MeshD mediates interaction between a host with Dr.Web for UNIX Internet Gateways installed and other cloud hosts.

Connection types

Dr.Web MeshD uses the following connection types:

- *Client (service)*—used by Dr.Web MeshD to connect to other cloud hosts. These hosts are clients of services provided by the given host.



Components of Dr.Web for UNIX Internet Gateways operating on the host and accessing services provided by the cloud through Dr.Web MeshD operating on the same host connect to it via a local UNIX socket. At that, a client connection is not used.

- *Partner (peer to peer)*—used by Dr.Web MeshD for interaction with peer (within a service) partner cloud hosts. Usually such horizontal connections are used for scaling and distributing the load when interacting with the cloud, as well as for synchronization of cloud hosts.
- *Uplink*—used by Dr.Web MeshD to connect this host (client) to cloud hosts (service providers) (for example, distribution of virus databases updates, file transmission for scanning, and so on).



The use of all three types of connections is configured independently for different cloud services. At that, the same host can be configured as a server for processing client requests within a service (for example, providing fresh updates) and as a client within another service (for example, remote file scanning).

Within cloud, hosts perform authorized interaction via a secure SSH protocol, that is, all sides of interhost communication are always mutually authenticated. For the authentication, *host keys* are used according to [RFC 4251](#). The client connection from the local component is always considered as trusted.

Operation Modes

Dr.Web MeshD can work both in the daemon mode and run on requests from other Dr.Web for UNIX Internet Gateways components, located on the local host. If Dr.Web MeshD is configured to serve client connections (i.e. the `ListenAddress` parameter is not empty) and at least one of the services is activated, Dr.Web MeshD starts as a daemon and waits for connections from clients. Besides, Dr.Web MeshD can be activated on the local host upon request, for example, when executing the [command](#):

```
$ drweb-ctl update --local-cloud
```

If Dr.Web MeshD is not set to process client connections (the `ListenAddress` parameter is empty) and there is no request to Dr.Web MeshD during the period specified in the `IdleTimeLimit` parameter, the component exits automatically.

Services

Exchanging updates (Update)

The service allows the host to subscribe to updates of virus and other databases, send notifications on the updates, upload and share the update files between cloud hosts. The service settings can be configured using the `Update*` parameters.

The standard service usage assumes that Dr.Web MeshD is installed with the enabled feature of obtaining updates on several machines (clients of the service) in the local network of the company. The typical client [settings](#) are:

```
...
[MeshD]
UpdateChannel = On
UpdateUplink = <server address>
ListenAddress =
...
[Update]
UseLocalCloud = Yes
...
```



On the local server for distributing updates, the following settings are specified:

```
UpdateChannel = On
UpdateUplink =
ListenAddress = <address>:<port>
```

Here, *<server address>* in the uplink connection of the client must refer to the *<address>* and *<port>* that are used by the server host for managing client connections.

When one of the hosts is being updated from the update servers (external to the local cloud—Dr.Web GUS update servers or the centralized protection server), the host sends a notification to all necessary clients (if the host is set as an update exchange server) and sends to the server host a new list of files available for distribution from the host. Upon receiving the notification, client hosts can request the update download from the server, that in turn can request the files from the client to save them locally or to send them to the other client that has the requested the files from the server.

The scenario decreases the update delay because clients send requests to Dr.Web GUS at different times, at thus the first updated client immediately distributes the fresh update files to all necessary cloud hosts. It also decreases amount of traffic and Dr.Web GUS load.



Note that when using the local cloud for update distribution, in addition to Dr.Web MeshD, hosts must contain the Dr.Web Updater component.

Remote file scanning (Engine)

The service allows to use Dr.Web Scanning Engine for scanning remote files: client hosts send files for scanning to the server host and server hosts provide a service for file scanning. The typical client [settings](#) are:

```
...
[MeshD]
EngineChannel = On
EngineUplink = <server address>
ListenAddress =
...
```

On the local scanning server, the following settings are specified:

```
EngineChannel = On
EngineUplink =
ListenAddress = <address>:<port>
```

Here, *<server address>* in the uplink connection of the client must refer to the *<address>* and *<port>* that are used by the server host for managing client connections.



Transmitting files for scanning (File)

The feature is not used (remote scanning is provided within the *Engine* service).

URL check

The service allows to use the server hosts to check URL for belonging to potentially dangerous and non-recommended categories: client hosts send URL to be checked to the server host. The typical client [settings](#) are:

```
...
[MeshD]
UrlChannel = On
UrlUplink = <server address>
ListenAddress =
...
```

On the local server used for URL check, the following settings are specified:

```
UrlChannel = On
UrlUplink =
ListenAddress = <address> : <port>
```

Here, <server address> in the uplink connection of the client must refer to the <address> and <port> that are used by the server host for managing client connections.



Command-Line Arguments

To run Dr.Web MeshD, type the following command in the command line:

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

Dr.Web MeshD can process the following options:

Parameter	Description
<code>--help</code>	Function: Output short help information about command-line parameters to the console or to the terminal emulator and to exit upon completion. Short form: <code>-h</code> Arguments: None.
<code>--version</code>	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: <code>-v</code> Arguments: None.

Example:

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

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

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 when needed by the [Dr.Web ConfigD](#) configuration daemon. 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-meshd`.

Configuration Parameters

The component uses configuration parameters which can be found in the `[MeshD]` section of the integrated [configuration file](#) of Dr.Web for UNIX Internet Gateways.



The section contains the following parameters:

Parameter	Description
LogLevel <i>{logging level}</i>	<p>Logging level of the component.</p> <p>If the parameter value is not specified, the <code>DefaultLogLevel</code> parameter value from the <code>[Root]</code> section is used.</p> <p>Default value: <code>Notice</code></p>
Log <i>{log type}</i>	<p>Logging method of the component.</p> <p>Default value: <code>Auto</code></p>
ExePath <i>{path to file}</i>	<p>Executable path to the component.</p> <p>Default value: <code><opt_dir>/bin/drweb-meshd</code>.</p> <ul style="list-style-type: none">• For GNU/Linux: <code>/opt/drweb.com/bin/drweb-meshd</code>.• For FreeBSD: <code>/usr/local/libexec/drweb.com/bin/drweb-meshd</code>
DebugSsh <i>{Boolean}</i>	<p>Performing logging of the SSH protocol messages (used for transferring messages and data) received and sent by Dr.Web MeshD operating on the host, if logging level is <code>LogLevel = Debug</code>.</p> <p>Default value: <code>No</code></p>
IdleTimeLimit <i>{time interval}</i>	<p>Maximum idle time for the component. When the specified period of time expires, the component shuts down.</p> <p>Acceptable values: from 10 seconds (10s) to 30 days (30d) inclusive. If the <code>None</code> value is set, the component will functionate eternally; the <code>SIGTERM</code> signal will not be sent if the component goes idle.</p> <p>Default value: <code>30s</code></p>
DnsResolverConfPath <i>{path to file}</i>	<p>Path to the subsystem configuration file of domain name permissions (DNS resolver).</p> <p>Default value: <code>/etc/resolv.conf</code></p>
ListenAddress <i><IP address>:<port></i>	<p>Network socket (address and port) of the client connection where the component is waiting to receive the connections from cloud hosts. These hosts are clients of services provided by this cloud host.</p> <p>In order for the component to listen on the IPv6 interface and detect cloud client hosts via IPv6, the parameter must be set. If the value is not specified, the component does not receive requests from clients.</p> <p>Default value: <i>(not set)</i></p>
UpdateChannel <i>{On Off}</i>	<p>Enabling or disabling the Dr.Web MeshD component working on this host, exchanging updates of virus databases between hosts of the cloud (for example, getting updates of virus databases from other cloud hosts and sending new updates to the cloud).</p>



Parameter	Description
	<p>If the parameter is set to On, the component Dr.Web MeshD is automatically started by the Dr.Web ConfigD configuration daemon.</p> <p>Default value: On</p>
UpdateUplink {address}	<p>Address of the upper host of Dr.Web MeshD that acts as a server providing URL checking services for this host.</p> <p>Allowed values:</p> <ul style="list-style-type: none">• <i>value is not specified</i>—server is not set for the service and Dr.Web MeshD will not connect to anywhere;• <i><IP address>:<port></i>—Dr.Web MeshD will connect to the server with the specified address and port;• <i>dns : <service name> [: <domain>]</i>—address and port of the server is determined by searching for SRV record of the <i><domain></i> DNS domain. If <i><domain></i> is not specified, a domain from the DNS resolver configuration file is used (the path is specified in <i>ResolverConfPath</i>). The domain is taken from the <i>search</i> or <i>domain</i> field, depending on which of them is encountered last;• <i>discover</i>—search for the higher host by the <i>discovery</i> mechanism. <p>Default value: (not specified)</p>
UpdateDebugIpc {Boolean}	<p>Output the debug information to the log for the update exchange service if logging level is <i>LogLevel = Debug</i>.</p> <p>Default value: No</p>
UpdateTraceContent {Boolean}	<p>Output the transmitted data to the log for the update exchange service if logging level is <i>LogLevel = Debug</i>.</p> <p>Default value: No</p>
FileChannel {On Off}	<p>Enable or disable an option that allows the Dr.Web MeshD component that works on this host to participate in exchanging files.</p> <p>If the parameter is set to On, the component Dr.Web MeshD is automatically started by the Dr.Web ConfigD configuration daemon.</p> <p>Default value: On</p>
FileUplink {address}	<p>Address of the higher host of Dr.Web MeshD that acts as a server which scans files from this host.</p> <p>Allowed values:</p> <ul style="list-style-type: none">• <i>value is not specified</i>—server is not set for the service and Dr.Web MeshD will not connect to anywhere;• <i><IP address>:<port></i>—Dr.Web MeshD will connect to the server with the specified address and port;• <i>dns : <service name> [: <domain>]</i>—address and port of the server is determined by searching for SRV record of the <i><domain></i> DNS domain. If



Parameter	Description
	<p><domain> is not specified, a domain from the DNS resolver configuration file is used (the path is specified in <code>ResolverConfPath</code>). The domain is taken from the <code>search</code> or <code>domain</code> field, depending on which of them is encountered last;</p> <ul style="list-style-type: none">• <code>discover</code>—search for the higher host by the <i>discovery</i> mechanism. <p>Default value: <i>(not specified)</i></p>
<code>FileDebugIpc</code> {Boolean}	<p>Output the debug information to the log for the file exchange service if logging level is <code>LogLevel = Debug</code>.</p> <p>Default value: <code>No</code></p>
<code>EngineChannel</code> {On Off}	<p>Enable or disable an option that allows the Dr.Web MeshD component that works on this host to participate providing scan engine services.</p> <p>If the parameter is set to <code>On</code>, the component Dr.Web MeshD is automatically started by the Dr.Web ConfigD configuration daemon.</p> <p>Default value: <code>On</code></p>
<code>EngineUplink</code> {address}	<p>Address of the higher host of Dr.Web MeshD that acts as a scanning server which provides the scan engine services for this host.</p> <p>Allowed values:</p> <ul style="list-style-type: none">• <i>value is not specified</i>—server is not set for the service and Dr.Web MeshD will not connect to anywhere;• <code><IP address>:<port></code>—Dr.Web MeshD will connect to the server with the specified address and port;• <code>dns:<service name> [: <domain>]</code>—address and port of the server is determined by searching for SRV record of the <domain> DNS domain. If <domain> is not specified, a domain from the DNS resolver configuration file is used (the path is specified in <code>ResolverConfPath</code>). The domain is taken from the <code>search</code> or <code>domain</code> field, depending on which of them is encountered last;• <code>discover</code>—search for the higher host by the <i>discovery</i> mechanism. <p>Default value: <i>(not specified)</i></p>
<code>EngineDebugIpc</code> {Boolean}	<p>Output the debug information to the log for the scan service if logging level is <code>LogLevel = Debug</code>.</p> <p>Default value: <code>No</code></p>
<code>UrlChannel</code> {On Off}	<p>Enable or disable an option that allows the Dr.Web MeshD component that works on this host to participate providing URL check services.</p> <p>Default value: <code>On</code></p>
<code>UrlUplink</code> {address}	<p>Address of the higher host of Dr.Web MeshD that acts as a server which provides the URL check services for this host.</p>



Parameter	Description
	<p>Allowed values:</p> <ul style="list-style-type: none">• <i>value is not specified</i>—server is not set for the service and Dr.Web MeshD will not connect to anywhere;• <code><IP address>:<port></code>—Dr.Web MeshD will connect to the server with the specified address and port;• <code>dns : <service name> [: <domain>]</code>—address and port of the server is determined by searching for SRV record of the <code><domain></code> DNS domain. If <code><domain></code> is not specified, a domain from the DNS resolver configuration file is used (the path is specified in <code>ResolverConfPath</code>). The domain is taken from the <code>search</code> or <code>domain</code> field, depending on which of them is encountered last;• <code>discover</code>—search for the higher host by the <i>discovery</i> mechanism. <p>Default value: <i>(not specified)</i></p>
<code>DiscoveryResponderPort</code> <i>{port number}</i>	<p>The port on which the higher host responds to the requests of the clients set via UDP protocol.</p> <p>The <i>discovery</i> mechanism is activated only if the <code>ListenAddress</code> value is set.</p> <p>Default value: 18008</p>
<code>UrlDebugIpc</code> <i>{Boolean}</i>	<p>Output the debug information to the log for the URL check service if logging level is <code>LogLevel = Debug</code>.</p> <p>Default value: No</p>



The current version of Dr.Web for UNIX Internet Gateways, the *File* file-transmitting service is not used. Use the *Engine* scan engine service instead.

Dr.Web URL Checker

Dr.Web URL Checker is an auxiliary component used by other component to check URLs for belonging to unwanted or malicious categories.

Dr.Web URL Checker is used by the following components:

- [Dr.Web HTTPD](#),
- [Dr.Web MeshD](#),
- [SpIDer Gate](#),
- [Dr.Web ICAPD](#)

Operating Principles

The Dr.Web URL Checker component is used by other components in order to check URLs for belonging to unwanted or potentially dangerous categories.



The check can be performed either by using specialized link bases or by using the Dr.Web CloudD service. To use the Dr.Web CloudD service, perform the following command:

```
$ drweb-ctl cfset Root.UseCloud Yes
```

The Dr.Web URL Checker cannot be launched by the user. It is launched by the Dr.Web ConfigD configuration management daemon upon request of other components.

Command-Line Arguments

To run Dr.Web URL Checker, type the following command in the command line:

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

Dr.Web URL Checker can process the following options:

Parameter	Description
--help	Function: 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: 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-urlcheck --help
```

This command outputs short help information on URL Checker.

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 when needed by the [Dr.Web ConfigD](#) configuration daemon. 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-urlcheck`.



Configuration Parameters

The component uses configuration parameters which can be found in the [Urlcheck] section of the integrated [configuration file](#) of Dr.Web for UNIX Internet Gateways.

The section contains the following parameters:

Parameter	Description
LogLevel <i>{logging level}</i>	<p>Logging level of the component.</p> <p>If the parameter value is not specified, the DefaultLogLevel parameter value from the [Root] section is used.</p> <p>Default value: Notice</p>
Log <i>{log type}</i>	<p>Logging method of the component.</p> <p>Default value: Auto</p>
ExePath <i>{path to file}</i>	<p>Executable path to the component.</p> <p>Default value: <opt_dir>/bin/drweb-urlcheck.</p> <ul style="list-style-type: none">• For GNU/Linux: /opt/drweb.com/bin/drweb-urlcheck.• For FreeBSD: /usr/local/libexec/drweb.com/bin/drweb-urlcheck
RunAsUser <i>{UID user name}</i>	<p>The name of the user on whose behalf the component is 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.</p> <p>When a user name is not specified, the component operation terminates with an error after the startup.</p> <p>Default value: drweb</p>
IdleTimeLimit <i>{time interval}</i>	<p>Maximum idle time for the component. When the specified period of time expires, the component shuts down.</p> <p>Acceptable values: from 10 seconds (10s) to 30 days (30d) inclusive.</p> <p>If the None value is set, the component will functionate eternally; the SIGTERM signal will not be sent if the components goes idle.</p> <p>Default value: 30s</p>



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](#) scan engine and of the components monitoring the access to the internet.

Operating Principles

The component is designed to address to the Doctor Web 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 the Doctor Web predefined categories of web resources. Besides, the component periodically sends to the Dr.Web Cloud the statistics on infected files detection and information on operating system, on which Dr.Web for UNIX Internet Gateways is run.

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.

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 [SplDer 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 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](#) scan 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
<code>--help</code>	Function: Output short help information about command-line parameters to the console or to the terminal emulator and to exit upon completion. Short form: <code>-h</code>



	Arguments: None.
<code>--version</code>	Function: Output information about the version of this component to the console or to the terminal emulator and to exit after completion. Short form: <code>-v</code> Arguments: None.

Example:

```
$ /opt/drweb.com/bin/drweb-cloudd --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 can be found in the [CloudD] section of the unified [configuration file](#) of Dr.Web for UNIX Internet Gateways.

The section contains the following parameters:

Parameter	Description
<code>LogLevel</code> <i>{logging level}</i>	Logging level of the component. If the parameter value is not specified, the <code>DefaultLogLevel</code> parameter value from the [Root] section is used. Default value: <code>Notice</code>
<code>Log</code> <i>{log type}</i>	Logging method of the component. Default value: <code>Auto</code>
<code>ExePath</code> <i>{path to file}</i>	Executable path to the component. Default value: <code><opt_dir>/bin/drweb-cloudd</code> . <ul style="list-style-type: none">• For GNU/Linux: <code>/opt/drweb.com/bin/drweb-cloudd</code>.



Parameter	Description
	<ul style="list-style-type: none">• For FreeBSD: <code>/usr/local/libexec/drweb.com/bin/drweb-cloudd</code>
<code>RunAsUser</code> <i>{UID user name}</i>	<p>The name of the user on whose behalf the component is 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: <code>RunAsUser = name:123456.</code></p> <p>When a user name is not specified, the component operation terminates with an error after the startup.</p> <p>Default value: <code>drweb</code></p>
<code>IdleTimeLimit</code> <i>{time interval}</i>	<p>Maximum idle time for the component. When the specified period of time expires, the component shuts down.</p> <p>Acceptable values: from 10 seconds (10s) to 30 days (30d) inclusive. If the <code>None</code> value is set, the component will functionate eternally; the <code>SIGTERM</code> signal will not be sent if the components goes idle.</p> <p>Default value: <code>1h</code></p>
<code>PersistentCache</code> <i>{On Off}</i>	<p>Enable or disable saving of the cache of Dr.Web Cloud replies to the disk.</p> <p>Default value: <code>off</code></p>
<code>DebugSdk</code> <i>{Boolean}</i>	<p>Indicates whether detailed Dr.Web Cloud messages should be included into the log file on the debug level (<code>LogLevel = DEBUG</code>).</p> <p>Default value: <code>No</code></p>



Dr.Web LookupD

The Dr.Web LookupD component is designed to refer to external sources (text files, relational databases, directory services, supporting the LDAP protocol interaction) to retrieve data by using the LDAP protocol. The received data is used in rules according to which network connections are scanned (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 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 components.

Dr.Web LookupD supports referrals to the following data sources:

- Text files (in the *AllMatch*, the *Mask*, the *Regex*, the *Cidr* modes);
- Relational databases (MySQL, PostgreSQL, SQLite);
- Redis data storages;
- Directory services (Active Directory and others which provide access via LDAP, for example, OpenLDAP).

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 text files, relational database, network storages and directory services (like Active Directory) that support the LDAP protocol. The received data (for example, users' identifiers and rights) is transferred to Dr.Web for UNIX Internet Gateways components to be used in different rules for scans (for example, to allow a user to access a requested URL and so on).



This manual does not describe the operating principles of relational databases, the Redis storage, directory services and the LDAP protocol. If necessary, refer to the corresponding reference materials.

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).

Once the request on data reception from a certain component arrives, [Dr.Web ConfigD](#) configuration daemon launches the Dr.Web LookupD (if not launched), then the component performs the request from the requested data source and returns the response. Depending on



the request, the reply consists of a list of strings that satisfy a certain search criteria, retrieved from the data source according to a given search criteria, or a logical value (true or false) that indicates whether the search results contain strings that match the given condition.

In Dr.Web LookupD 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 Dr.Web for UNIX Internet Gateways 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.

Text files processing peculiarities

1. When processing text files, leading and trailing spaces in strings are discarded. Blank strings and lines that have the '#' character as the first non-whitespace character, are ignored.
2. Text files are considered immutable data sources and their content is fully cached in memory. In addition, the results of requests to these files for validation are also cached. Thus, in case if the source file has been modified, it is needed to make Dr.Web LookupD re-read the configuration by sending HUP signal to Dr.Web ConfigD component, using, for example, `reload` command of `drweb-ctl`.

MySQL connection aspects

Before the MySQL connection, the parameters from the `[client]` section of the MySQL file settings are read by default (the file search is done in the following paths: `/etc/my.cnf`, `/etc/mysql/my.cnf`, and `/etc/alternatives/my.cnf`).

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
<code>--help</code>	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: <code>-h</code> Arguments: None.
<code>--version</code>	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: <code>-v</code>



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

In this section

- [Component Parameters](#)
- [Data Source Sections](#)
- [Adding sections for new data sources](#)

The component uses configuration parameters which can be found in the `[LookupD]` section of the integrated [configuration file](#) of Dr.Web for UNIX Internet Gateways.

Component Parameters

The section contains the following parameters:

Parameter	Description
<code>LogLevel</code> <i>{logging level}</i>	Logging level of the component. If the parameter value is not specified, the <code>DefaultLogLevel</code> parameter value from the <code>[Root]</code> section is used. Default value: <code>Notice</code>
<code>Log</code> <i>{log type}</i>	Logging method of the component. Default value: <code>Auto</code>



Parameter	Description
<code>ExePath</code> <i>{path to file}</i>	<p>Executable path to the component.</p> <p>Default value: <code><opt_dir>/bin/drweb-lookupd</code>.</p> <ul style="list-style-type: none">• For GNU/Linux: <code>/opt/drweb.com/bin/drweb-lookupd</code>.• For FreeBSD: <code>/usr/local/libexec/drweb.com/bin/drweb-lookupd</code>
<code>RunAsUser</code> <i>{UID user name}</i>	<p>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: <code>RunAsUser = name:123456</code>.</p> <p>When a user name is not specified, the component operation terminates with an error after the startup.</p> <p>Default value: <code>drweb</code></p>
<code>IdleTimeLimit</code> <i>{time interval}</i>	<p>Maximum idle time for the component. When the specified period of time expires, the component shuts down.</p> <p>Acceptable values: from 10 seconds (10s) to 30 days (30d) inclusive. If the <code>None</code> value is set, the component will functionate eternally; the <code>SIGTERM</code> signal will not be sent if the components goes idle.</p> <p>Default value: <code>30s</code></p>
<code>DebugLibldap</code> <i>{Boolean}</i>	<p>Indicates whether debug messages of the <code>libldap</code> library are also included into the log file on the debug level (i.e. when <code>LogLevel = DEBUG</code>).</p> <p>Default value: <code>No</code></p>
<code>LdapCheckCertificate</code> <i>{No Allow Try Yes}</i>	<p>The mode of certificate verification for LDAP connections via SSL/TLS.</p> <p>Allowed values:</p> <ul style="list-style-type: none">• <code>No</code>—do not request the server certificate;• <code>Allow</code>—request the server certificate. If the certificate is not provided, the session will continue in its normal way. If the server certificate is provided but cannot be scanned (it is impossible to find the corresponding root certificate), the certificate will be ignored and the session keeps running in its normal way;• <code>Try</code>—request the server certificate. If the certificate is not provided, the will continue in its normal way. If the server certificate is provided but cannot be checked (it is impossible to find the corresponding root certificate), the session will be terminated;• <code>Yes</code>—request the server certificate. If the certificate is not provided or cannot be scanned (it is impossible to find the corresponding root certificate), the session is terminated.



Parameter	Description
	<p>For LDAP data sources this certificate verification mode influences the way the URL is processed when the <code>ldaps://</code> scheme or the StartTLS extension is used; and for AD data sources it will influence connections to the server, if <code>UseSSL=Yes</code> has been specified in the corresponding section (see below).</p> <p>Default value: <code>Yes</code></p>
<code>LdapCertificatePath</code> {path to file}	<p>Path to the SSL certificate used for connection to the LDAP servers (Active Directory) via a secure SSL/TLS connection.</p> <p>Please note that the certificate file and the private key file (which is specified by a parameter described below) must form a matching pair.</p> <p>Default value: (not set)</p>
<code>LdapKeyPath</code> {path to file}	<p>Path to the private key used for connection to the LDAP servers (Active Directory) via a secure SSL/TLS connection.</p> <p>Please note that the certificate file and the private key file (which is specified by the mentioned parameter) must form a matching pair.</p> <p>Default value: (not set)</p>
<code>LdapCaPath</code> {path}	<p>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.</p> <p>Default value: <code><path to the list of trusted certificates></code>. The path depends on your GNU/Linux distribution.</p> <ul style="list-style-type: none">• For Astra Linux, Debian, Linux Mint, SUSE Linux and Ubuntu, usually it is a path <code>/etc/ssl/certs/</code>.• For CentOS and Fedora—a path <code>/etc/pki/tls/certs/ca-bundle.crt</code>.• For other distributions a path can be defined through results of execution of the command <code>openssl version -d</code>.• If a command is unavailable or an OS distribution could not be identified, the value <code>/etc/ssl/certs/</code> is used.
<code>DbIdleTimeout</code> {time interval}	<p>The time-out period at the end of which the established connection to the database (or the Redis storage) will be broken if it is idle.</p> <p>Default value: <code>5m</code></p>
<code>MysqlDefaultConn</code> {URL}	<p>The URI that sets the parameters for connecting to the MySQL database by default.</p> <p>Allowed values:</p> <ul style="list-style-type: none">• <code>tcp://[<user>[:<password>]@][<host>][: <port>][/ <database name>][?<parameter>=<value>[&...]]</code>;• <code>unix://[<user>[:<password>]@]<path to socket>[: <database name>][?<parameter>=<value>[&...]]</code>.



Parameter	Description
	Note the URI requirements . Default value: <i>(not set)</i>
<code>PgDefaultConn</code> <i>{URL}</i>	The URI that sets the parameters for connecting to the PostgreSQL database by default. Allowed values: <ul style="list-style-type: none"><code>tcp://[<user>[:<password>]@][<host>][:<port>][/<database name>][?<parameter>=<value>[&...]]</code>;<code>unix://[<user>[:<password>]@]<path to socket>[:<database name>][?<parameter>=<value>[&...]]</code>. Note the URI requirements . Default value: <i>(not set)</i>
<code>SqliteDefaultConn</code> <i>{path to file}</i>	Path to a default SQLite database file (specify the <code>file://</code> scheme prefix). Default value: <i>(not set)</i>
<code>RedisDefaultConn</code> <i>{URL}</i>	The URL that sets the connection parameters for the Redis database by default. Allowed values: <ul style="list-style-type: none"><code>tcp://[<password>@][<host>][:<port>][/<database index>]</code>;<code>unix://[<password>@]<socket path>[:<database index>]</code>. Note the URI requirements . Default value: <i>(not set)</i>

URI requirements for database connection

1. Use only the `tcp:` and `unix:` scheme prefixes (for local UNIX sockets). database-specific prefixes (such as `postgresql:` and `mysql:`) are not supported. Path to a SQLite database file is specified with the `file://` scheme prefix.
2. If the `<host>` field is not specified in URI or the `localhost` host is specified, the `127.0.0.1` host address is substituted. In this case for the MySQL and PostgreSQL databases, the connection is established by default *via a local UNIX socket* in spite that a network connection is specified.
3. If URI fields (such as `<user>`, `<password>`, `<database name>`, and so on) or the connection-parameters string contain special characters (space, column, and so on), use hex coding, for example:
 - space—"`%20`";
 - `:`—"`%3A`";
 - `'`—"`%2F`";



- '@'—"%40";
- '% '—"%25".

4. For MySQL, the connection parameter string can only include the following parameters:

Parameter name	Convention in database documents	Type	Description
init	<i>MYSQL_INIT_COMMAND</i>	String	A SQL command to be executed after connecting to the database
compression	<i>MYSQL_OPT_COMPRESS</i>	Logical	Use data compression
connect-timeout	<i>MYSQL_OPT_CONNECT_TIMEOUT</i>	Integer	Time-out for disconnecting an unused connection in seconds
reconnect	<i>MYSQL_OPT_RECONNECT</i>	Logical	Allow or deny automatic reconnection
read-timeout	<i>MYSQL_OPT_READ_TIMEOUT</i>	Integer	Time-out for receiving packets from server in seconds
write-timeout	<i>MYSQL_OPT_WRITE_TIMEOUT</i>	Integer	Time-out for sending packets to the server in seconds
charset	<i>MYSQL_SET_CHARSET_NAME</i>	String	Name of the character encoding used for the default connection
plugin-dir	<i>MYSQL_PLUGIN_DIR</i>	String	Path to the directory on the server storing the plug-ins
nonblock	<i>MYSQL_OPT_NONBLOCK</i>	Integer	Stack size for non-blocking I/O operations
ssl-key	<i>MYSQL_OPT_SSL_KEY</i>	String	Path to the private key (in PEM format) used to establish a secure connection
ssl-cert	<i>MYSQL_OPT_SSL_CERT</i>	String	Path to the public key certificate (in PEM format) used to establish a secure connection



Parameter name	Convention in database documents	Type	Description
<code>ssl-ca</code>	<code>MYSQL_OPT_SSL_CA</code>	String	Path to the file (in PEM format) containing trusted CA certificates
<code>ssl-capath</code>	<code>MYSQL_OPT_SSL_CAPATH</code>	String	Path to the directory containing trusted CA certificates (in PEM format)
<code>ssl-cipher</code>	<code>MYSQL_OPT_SSL_CIPHER</code>	String	List of supported encryption algorithms for secure connection
<code>ssl-crl</code>	<code>MYSQL_OPT_SSL_CRL</code>	String	Path to the file (in PEM format) containing revoked certificates
<code>ssl-crlpath</code>	<code>MYSQL_OPT_SSL_CRLPATH</code>	String	Path to the directory containing revoked certificates (in PEM format)
<code>ssl-fp</code>	<code>MARIADB_OPT_SSL_FP</code>	String	the SHA1 hash of the valid server certificate
<code>ssl-fp-list</code>	<code>MARIADB_OPT_SSL_FP_LIST</code>	String	Path to the file containing the SHA1 hashes of valid server certificates
<code>tls-passphrase</code>	<code>MARIADB_OPT_TLS_PASSPHRASE</code>	String	Password for the password-protected client private key
<code>tls-version</code>	<code>MARIADB_OPT_TLS_VERSION</code>	String	List of supported TLS versions
<code>server-verify-cert</code>	<code>MYSQL_OPT_SSL_VERIFY_SERVER_CERT</code>	Logical	Allow or deny verification of server certificates
<code>server-public-key-path</code>	<code>MYSQL_SERVER_PUBLIC_KEY</code>	String	Path to the file (in PEM format) containing the RSA server public key

Read more on these parameters in the documents to the database:
https://mariadb.com/kb/en/mysql_options/.



5. For the PostgreSQL database also refer to <https://www.postgresql.org/docs/current/libpq-connect.html#LIBPQ-PARAMKEYWORDS>.

Data Source Sections

In addition to the general section [LookupD], the configuration file also contains sections that describe connections to data sources (one section for each connection). These sections are named using the following scheme: [LookupD.<type>.<name>], where:

- <type>—connection type:
 - LDAP—for directory service that uses LDAP;
 - AD—Active Directory service;
 - AllMatch—for text file in the *AllMatch* mode (full identity);
 - Mask—for text file in the *Mask* mode (mask identity);
 - Regex—for text file in the *Regex* mode (identity to a regular expression in PCRE standard);
 - Cidr—for text file in the *Cidr* mode (IP addresses or IP address ranges identity);
 - Pq—for the PostgreSQL database;
 - Mysql—for the MySQL database;
 - Sqlite—for the SQLite database;
 - Redis—for the Redis database;
- <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

Parameter	Description
Url {string}	<p>URL that defines the used LDAP server and extracted data. According to RFC 4516, URL is built on the basis of the following scheme:</p> <pre><scheme> : // <host>[:<port>] / <dn>[?<attrs>[?<scope>[?<filter>[?<extensions>]]]]</pre> <p>Where:</p> <ul style="list-style-type: none"> <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;



	<p><scope>—search scope (base, one, sub);</p> <p><filter>—filtering condition for values of extracted attributes;</p> <p><extensions>—list of LDAP extensions used in the request.</p> <p>Features</p> <ul style="list-style-type: none"> • 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: <ul style="list-style-type: none"> ▫ \$u is automatically replaced with <code>user</code>, the user name, sent by the client component; ▫ \$d is automatically replaced with <code>domain</code>, the domain, sent by the client component; ▫ \$D—chain <subdomain>.<domain>, modified into <code>dc=<subdomain>,dc=<domain></code>; ▫ \$\$—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 <code>\XX</code>. Besides, special characters in URL LDAP are encoded using sequences <code>%XX</code>. For example, when using URL according to the scheme <code>ldapi</code> of the character '/' as a part of the path to the local LDAP server socket, this character is encoded as <code>%2f</code>. • As allowed extensions in <extensions>, only <code>StartTLS</code> and <code>1.3.6.1.4.1.1466.20037</code> 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 <code>ldaps</code>) If the name of the used extension is preceded by the character '!', then usage of TLS is <i>required</i>, i.e. in case the establishment of the secure connection is impossible, the request <i>will not</i> be handled. Otherwise, the request will be handled even if the secure connection is not established. <div style="background-color: #e6f2e6; padding: 10px; border: 1px solid #ccc; margin: 10px 0;">  <p style="margin: 0;">Indicated extensions could not be used with the protected <code>ldaps</code> scheme. For more information refer to RFC 4516 or <code>man ldap_search_ext_s</code>.</p> </div> <p>Examples:</p> <pre>"ldaps://ds.example.com:990/\$D?givenName,sn,cn?sub?(uid=\$u)" "ldap://ldap.local/o=org,dc=nodomain?ipNetworkNumber?sub?(objectClass=ipNetwork)?!StartTLS"</pre> <p>Default value: <i>(not set)</i></p>
BindDn	An object in the LDAP directory to which the user is bound to get authorization.



<i>{string}</i>	Example: "cn=admin,dc=nodomain". Default value: <i>(not set)</i>
BindPassword <i>{string}</i>	The user's password for authentication on the LDAP server. Default value: <i>(not set)</i>
ChaseReferrals <i>{Boolean}</i>	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

Parameter	Description
Host <i>{string}</i>	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: <i>(not set)</i>
Port <i>{integer}</i>	Port on the host which is listened to by the server of the Active Directory service. Default value: 389
Dn <i>{string}</i>	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: <i>(not set)</i>
User <i>{string}</i>	The full identifier of a user on the server, to be used for identification. Example: "Administrator@WIN.LOCAL". Default value: <i>(not set)</i>
Password <i>{string}</i>	Password of the user for authentication on the Active Directory server. Default value: <i>(not set)</i>
ChaseReferrals <i>{Boolean}</i>	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 <i>{Boolean}</i>	Instructs to use SSL/TLS for connecting to the Active Directory. Default value: No



3. Section parameters of AllMatch, Mask, Regex, Cidr types

Parameter	Description
File	Path to a text file containing search strings.
{path}	Example: <code>"/etc/file1"</code> .
	Default value: <i>(not set)</i>

Features

- Strings from a file, specified in a section of `AllMatch` type, are used for the case-insensitive search for the exact string match.
- Strings of a file, specified in a section of the `Mask` type, are considered as masks (*wildcards*). Masks can be considered as a simplified version of regular expressions that contain standard and special characters. Matching the strings with the masks is implemented case-insensitively. Masks can contain the following special characters and expressions:

*—any character sequence;

?—any one symbol;

[<character set>]—a character from the set (for example, [bac]);

[<character set>]—a character that does not match any symbol from the set (for example, [!cab]);

[[:<class>:]]—a character from the POSIX class of characters (*alnum, alpha, ascii, blank, cntrl, digit, graph, lower, print, punct, space, upper, xdigit*).

A mask that matches a substring must contain the substring surrounded by the "*" symbols (e.g., *host*). If you need to specify one of the special characters, you need to escape character with the backslash: \[, \], *, \?. If needed, the backslash can be escaped as well: \\. Escape of any other characters does not make any sense, e.g. the string `\a\b\c*\d?\` will be converted to the `abc*d?\` string. Mask examples:



```
#Matches the "name" string exactly
name

#Matches the three-character strings where
#the first character is "c", the second is any, and the third is "t"
#For example: "cat", "cut", "cct"
c?t

#Matches the strings: "user", "users", "us3rr", "ussrl", and so on
#(the [:alpha:] character class matches any alphabetical
#character, the special character "?" matches any character)
us[[:alpha:]]34]r?

#Matches the strings: ".con", "file.col", "3...co!", and so on
#(any character sequence before the ".co", after-
#any character except "m" and "?")
*.co[!m\?]
```

- Strings from file, specified in section of the `Regex` type, are interpreted as PCRE (*Perl Compatible Regular Expressions*) regular extensions. Matching the strings with the regular expressions is implemented case-insensitively. Examples of regular expressions:

```
#IPv4
(\d{1,3}\.){3}\d{1,3}

#Email address in the .com domain
\w+@\w+\.com
```

- Strings from file, specified in section of the `Cidr` type, are interpreted as IP addresses or IP address ranges. IPv4 and IPv6 formats for IP addresses as well as IP address ranges are allowed. The subnet mask can be specified in bit (octet) format, as well as in CIDR (*Classless Inter-Domain Routing*) notation. For example:

```
#IPv4
192.168.0.1
192.168.0.0/12
192.168.0.0/255.255.255.224

#IPv6
fe80::c7e8/32
fe80::c7e8/255.255.255.224
```

4. Parameters used in sections of the `Pq`, `Mysql`, `Sqlite` type

Conn <i>{string}</i>	Database connection string. Allowed values: <ul style="list-style-type: none">• for the <code>Mysql</code> (MySQL), <code>Pq</code> (PostgreSQL) section:
-------------------------	--



	<pre>tcp://[<user>[:<password>]@]<host>[:<port>][/<database name>][?<parameter>=<value>[&...]]</pre> <pre>unix://[<user>[:<password>]@]<path to socket>[:<database name>][?<parameter>=<value>[&...]]</pre> <p>Example: "tcp://user:pwd@localhost:1234/userdb", "unix://user:pwd@/tmp/pgsql.sock:userdb"</p> <p>Note the URI requirements.</p> <ul style="list-style-type: none"> for the <code>Sqlite</code> (SQLite) section: <ul style="list-style-type: none"> Path to a database file (specify the <code>file://</code> scheme prefix). Example: "file:///home/user/users.db" <p>Default value: <i>defined by the corresponding *DefaultConn parameter value</i></p>
Request {string}	<p>SQL query string (<code>SELECT</code>) to a database. As for sources of <code>AD</code> and <code>LDAP</code> types, in query, the following automatically permitted markers can be used:</p> <ul style="list-style-type: none"> <code>\$u</code>, <code>\$U</code> is automatically replaced with <code>user</code>, the user name sent by the client component; <code>\$d</code>, <code>\$D</code> is automatically replaced with <code>domain</code>, the domain sent by the client component; <code>\$\$</code> is replaced with <code>'\$'</code> character. <p>Example: "SELECT username FROM users INNER JOIN domains ON users.domain = domains.id WHERE domains.name = \$d AND users.name = \$u"</p> <p>Default value: <i>(not set)</i></p>



As an SQL query, only query of the `SELECT` type can be specified. After implementing the substitutions, the query is transmitted to the database "as is". If the query result contains more than one column, then all columns except the first one will be ignored.

5. Parameters used in sections of the Redis type

Conn {string}	<p>Connection string with the Redis data storage.</p> <p>Allowed values:</p> <ul style="list-style-type: none"> <code>tcp://[<password>@]<host>[:<port>][/<database index>]</code> <code>unix://[<password>@]<socket path>[:<database index>]</code> <p>Note the URI requirements.</p> <p>Example: "tcp://localhost:6379"</p> <p>Default value: <i>defined by RedisDefaultConn parameter value</i></p>
Request {string}	<p>Redis storage query string. In query, the following automatically permitted markers can be used:</p>



	<ul style="list-style-type: none">• \$u, \$U is automatically replaced with <code>user</code>, the user name sent by the client component;• \$d, \$D is automatically replaced with <code>domain</code>, the domain sent by the client component;• \$\$ is replaced with '\$' character. <p>Example: "HVALS bad_users"</p> <p>Default value: <i>(not set)</i></p>
--	---



If the query result contains more than one column, then all columns except the first one will be ignored.

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 with 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 run the following [command](#):

```
# drweb-ctl reload
```



Dr.Web StatD

The Dr.Web StatD component is designed for accumulating statistics of events that occur during the operation of Dr.Web for UNIX Internet Gateways components. The events are stored in the permanent repository and can be obtained on request.

Operating Principles

The component ensures accumulation and permanent storage of events obtained during the operation of Dr.Web for UNIX Internet Gateways components. The following types of events are logged:

- Component emergency shutdown;
- Threat detection (in email messages, among other sources).

Dr.Web StatD works in the daemon mode and is automatically launched by the configuration control daemon. Event viewing and management is ensured by the `command` events of the [Dr.Web Ctl](#) utility.

Command-Line Arguments

To run Dr.Web StatD, type the following command in the command line:

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

Dr.Web StatD can process the following options:

Parameter	Description
<code>--help</code>	Function: Output short help information about command-line parameters to the console or to the terminal emulator and to exit upon completion. Short form: <code>-h</code> Arguments: None.
<code>--version</code>	Function: Output information about the version of this component to the console or to the terminal emulator and to exit after completion. Short form: <code>-v</code> Arguments: None.

Example:

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

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



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-statd`.

Configuration Parameters

The component uses configuration parameters which can be found in the `[StatD]` section of the integrated [configuration file](#) of Dr.Web for UNIX Internet Gateways.

The section contains the following parameters:

Parameter	Description
<code>LogLevel</code> <i>{logging level}</i>	Logging level of the component. If the parameter value is not specified, the <code>DefaultLogLevel</code> parameter value from the <code>[Root]</code> section is used. Default value: <code>Notice</code>
<code>Log</code> <i>{log type}</i>	Logging method of the component. Default value: <code>Auto</code>
<code>ExePath</code> <i>{path to file}</i>	Executable path to the file of the component. Default value: <code><opt_dir>/bin/drweb-statd</code> . <ul style="list-style-type: none">• For GNU/Linux: <code>/opt/drweb.com/bin/drweb-statd</code>.• For FreeBSD: <code>/usr/local/libexec/drweb.com/bin/drweb-statd</code>
<code>RunAsUser</code> <i>{UID user name}</i>	The name of the user on whose behalf the component is 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: <code>RunAsUser = name:123456</code> . When a user name is not specified, the component operation terminates with an error after the startup. Default value: <code>drweb</code>



Parameter	Description
<code>IdleTimeLimit</code> <i>{time interval}</i>	<p>Maximum idle time for the component. When the specified period of time expires, the component shuts down.</p> <p>Acceptable values: from 10 seconds (10s) to 30 days (30d) inclusive. If the <code>None</code> value is set, the component will functionate eternally; the <code>SIGTERM</code> signal will not be sent if the components goes idle.</p> <p>Default value: 30s</p>
<code>MaxEventStoreSize</code> <i>{size}</i>	<p>Maximum allowed size of the event database. Defined in <code>mb</code>, for example: <code>MaxEventStoreSize = 100mb</code>.</p> <p>Minimum value: 50mb.</p> <p>Default value: 1GB</p>



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, or saving 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 their 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 so on) 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, and so on).
- *chat worms* use protocols of popular messengers and chat programs (ICQ, IM, IRC, and so on).

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, and so on) or makes it possible for hackers to access the computer without permission, for example, to harm the computer of a third party.



A trojan 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, and so on).
- *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 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, and so on.

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

In this appendix

- [Detection Methods](#)
- [Threat-related Actions](#)

All Doctor Web anti-virus solutions use several malicious software detection methods simultaneously, and that allows them to perform thorough scans for 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 scanning 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 scans 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.

Cloud-based Threat Detection Technologies

Cloud-based detection methods allow to scan any object (file, application, browser extension, etc.) by its hash value. Hash is a unique sequence of numbers and letters of a given length. When analyzed by a hash value, objects are scanned using the existing database and then classified into categories: clean, suspicious, malicious, etc.

This technology optimizes the time of file scanning and saves device resources. The decision on whether the object is malicious is made almost instantly, because it is not the object that is analyzed, but its unique hash value. If there is no connection to the Dr.Web Cloud servers, the files are scanned locally, and the cloud scan resumes when the connection is restored.



Thus, the Dr.Web Cloud service collects information from numerous users and quickly updates data on previously unknown threats increasing the effectiveness of device protection.

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** (*Ignore*)—instructs to skip the detected threat without performing any other action.
- **Report** (*Report*)—instructs to inform on the detected threat without performing any other action.
- **Cure** (*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** (*Quarantine*)—instructs to move the detected threat to a special directory and isolate it from the rest of the system.
- **Delete** (*Delete*)—instructs to remove the infected object permanently.



If threat is detected in a file located in a container (an archive, email message, and so on), its removal is replaced with moving of a container to quarantine.

Appendix C. Technical Support

If you have a problem installing or using Doctor Web products, please try the following before contacting technical support:

- Download and review the latest manuals and guides at <https://download.drweb.com/doc/>.
- See the Frequently Asked Questions section at https://support.drweb.com/show_faq/.
- Browse the official Doctor Web forum at <https://forum.drweb.com/>.

If you haven't found a solution to your problem, you can request direct assistance from Doctor Web technical support specialists. Please use one of the options below:

- Fill out a web form in the appropriate section at <https://support.drweb.com/>.
- Call +7 (495) 789-45-86 (for customers in Moscow) or 8-800-333-79-32 (a toll-free line for customers within Russia).

For information on regional and international offices of Doctor Web, please visit the official website at <https://company.drweb.com/contacts/offices/>.

To facilitate processing of your issue, we recommend that you generate a data set for the installed product, its configuration, and system environment before contacting the technical support. To do that, you can use a special utility included in the Dr.Web for UNIX Internet



Gateways distribution.

To collect the data for technical support, use the command:

```
# <opt_dir>/bin/support-report.sh
```

where *<opt_dir>* is a directory for Dr.Web for UNIX Internet Gateways files, including executables and libraries (*/opt/drweb.com* by default for GNU/Linux). For details on conventions used for directories, refer to [Introduction](#).



To collect all data required for technical support, we recommend that you launch the utility with superuser privileges (i.e. privileges of the *root* user). To elevate your privileges, log in as a different user with the *su* command or use the *sudo* command to execute the command on behalf of another user.

During operation, the utility collects and archives the following information:

- OS data (name, architecture, result of the `uname -a` command);
- list of packages installed to your system, including Doctor Web packages;
- log contents:
 - Dr.Web for UNIX Internet Gateways logs (if configured for separate components);
 - log of the `syslog` system daemon (`/var/log/syslog`, `/var/log/messages`);
 - log of a system package manager (`apt`, `yum`, etc.);
 - the `dmesg` log;
- output of the commands: `df`, `ip a` (`ifconfig -a`), `ldconfig -p`, `iptables-save`, `nft export xml`;
- information on settings and configuration of Dr.Web for UNIX Internet Gateways:
 - list of downloaded virus databases (`drweb-ctl baseinfo -l`);
 - list of files from Dr.Web for UNIX Internet Gateways directories and MD5 hash values of these files;
 - Dr.Web Virus-Finding Engine scan engine version and MD5 hash value;
 - configuration parameters of Dr.Web for UNIX Internet Gateways (including contents of `drweb.ini`, `rules`, value files used in rules, Lua procedures, etc.);
 - user information and permissions retrieved from the key file, if Dr.Web for UNIX Internet Gateways is running in the standalone mode.

An archive containing information on the product and its system environment will be saved to the home directory of the user that launched the utility. The file will be named as follows:

```
drweb.report.<timestamp>.tgz
```

where *<timestamp>* is a full timestamp of creating the report, down to milliseconds, for



example: 20190618151718.23625.



Appendix D. Dr.Web for UNIX Internet Gateways Configuration File

Configuration parameters of all the 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 command:

```
$ drweb-ctl cfshow
```

You can change any parameter value in one of the two 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 run the following [command](#):

```
# drweb-ctl reload
```

2. Type this command in the command line:

```
# drweb-ctl cfset <section> .<parameter> <new value>
```



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.

- The 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 the Dr.Web for UNIX Internet Gateways component name, 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 the Dr.Web for UNIX Internet Gateways components while reading configuration parameters.
- Each of the lines in the file can contain only one parameter value:

```
<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.
- The order of sections in the file as well as the order of parameters inside the sections 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 take multiple 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.

You can specify multiple values as:

- 1) a comma-separated list:

```
Parameter = Value1, Value2, "Value 3"
```

- 2) a sequence of lines in the configuration file:

```
Parameter = Value2  
Parameter = Value1  
Parameter = "Value 3"
```

The order of values is arbitrary.



Path to files and directories are always enclosed into quotation marks when separated by commas, e.g:

```
ExcludedPaths = "/etc/file1", "/etc/file2"
```

If you represent as a set of path as a sequence of lines, quotation marks are not necessary:

```
ExcludedPaths = /etc/file1  
ExcludedPaths = /etc/file2
```

- If a parameter can take multiple values, it is indicated in the comments in the configuration file or in the text of the current manual.



For description of the configuration file sections, see description of the Dr.Web for UNIX Internet Gateways components.

Parameter Types

Configuration parameters can belong to the following types:

- *address*—network connection address specified as `<IP address>:<port>`;
- *boolean*—parameter having only two possible values: `Yes` or `No`;
- *integer*—a non-negative integer;
- *fractional number* — a non-negative number with a fractional part;
- *time interval*—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` or no suffix—seconds.

If the time interval is specified in seconds, you can 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: $N_1wN_2dN_3hN_4mN_5[N_6]s$, where N_1, \dots, 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 `drweb-ctl cfset` [command](#):

```
# drweb-ctl cfset Update.UpdateInterval 1802.5s
```

3) via a command-line parameter (for example, for [Command-Line Arguments](#)):

```
$ drweb-se --WatchdogInterval 1802.5
```



- *size*—parameter value can be the size of an object (file, buffer, cache, 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).



The file path must be ended with the file name.



In UNIX-like systems, names of directories 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 the Dr.Web for UNIX Internet Gateways component events are logged. The following values are possible:
 - *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 the Dr.Web for UNIX Internet Gateways component performs logging (its logging method). The following values are possible:
 - *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 `drweb-ctl cfset` [command](#):

```
# 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 the Dr.Web for UNIX Internet Gateways component 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

In this section

- [General Information](#)
- [Rule Format](#)
- [Conditions](#)



- [Actions](#)
- [Variables used in rules](#)
- [Categories of unwanted websites and threats](#)
- [Configuration parameters that can be used in rule conditions](#)
- [Features of saving rules to the configuration file](#)

General Information

The rules are represented by such constructions as IF *<conditional_part>* THEN *<action_part>*. At that, in the *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 *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".

The actions specified in a rule are executed only if the conditional part is true. If the conditional part is false, the actions specified in this rule are not performed, and the program jumps to the next rule. The rules are processed from the top down until an ultimate resolution is performed. After this, all rules below (if there are any) are ignored. When a rule is executed, it is important that actions in the *action part* be performed in order in which they are specified from left to right, and if there is an ultimate resolution in the chain of actions that interrupts the rule processing, the rest of the actions specified in the *action part* is not performed.

Rule Format

The rules have the format:

```
[<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. Key words, names of variables and configuration parameters used in the rules are not case-sensitive.



Conditions

The following types of conditions can be used in the rules:

Condition	Comment
<code><variable> <value ></code>	<p>The value of the variable coincides with that specified in the rule.</p> <p>Can be used only for variables that can contain a set of values simultaneously.</p>
<code><variable> [not] in <set of values></code>	<p>The value of the variable is contained in the specified set of values (for <code>not</code>—does not match any value from the specified set).</p>
<code><variable> [not] match <set of values></code>	<p>The value of the variable matches any regular expression listed in the specified set (for <code>not</code>—does not match any expression from the specified set).</p> <div style="border: 1px solid #ccc; background-color: #e6f2e6; padding: 10px; margin-top: 10px;">  <p>Regular expressions are specified using either the <i>POSIX</i> syntax (<i>BRE</i>, <i>ERE</i>) or the <i>Perl</i> syntax (<i>PCRE</i>, <i>PCRE2</i>).</p> </div>
<code><variable> [not] gt <value></code>	<p>The value of the variable is (not) greater than the value specified.</p> <p>Can be used only for those variables that can have a single value.</p>
<code><variable> [not] lt <value></code>	<p>The value of the variable is (not) less than the value specified.</p> <p>Can be used only for variables that have a single value.</p>

*) The 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
<code>(<value 1>[, <value 2>[, ...]])</code>	<p>In brackets the values to check against are listed (the minimum number must not be less than one). In case there is only one value and the <code>in</code> condition is used, you can omit the brackets (and you will end up with a case <code><variable> <value></code>).</p>



Syntax	Meaning
<code>"<section> . <parameter>"</code>	<p>The set of values currently assigned to a configuration parameter; the name of a configuration parameter whose value (or set of values) must be checked in quotation marks (note that you also need to specify the name of the section to which the parameter belongs).</p> <p>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.</p>
<code>file("<file name>")</code>	<p>Path to the text file <i><file name></i> with the list of values. Each string in the file corresponds to a list item. Leading and trailing spaces are ignored.</p> <p>You can use only absolute paths.</p> <p>If <i><file name></i> contains quotes and apostrophes, escape them: <code>'\'</code>.</p> <div data-bbox="916 902 1450 1832" style="background-color: #e6f2e6; padding: 10px;"><p> The file size must not exceed 64 MB.</p><p>The file content is read and inserted into the rules when the configuration file is being loaded. If the file does not exist or the file size is exceeded, an error x102 is returned when loading the settings.</p><p>If edit the file while Dr.Web for UNIX Internet Gateways is in operation, you need to reload it in order to apply the changes. Use the command:</p><pre data-bbox="1058 1496 1430 1570"># drweb-ctl reload</pre><p>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.</p></div>
<code><type_of_LOOKUP_request>@<tag> [@<value>]</code>	<p>A value array is requested via Dr.Web LookupD from an external data source, where <i><LOOKUP_query_type></i> is the type of source; <i><tag></i> is the name of the section describing the</p>



Syntax	Meaning
	<p>connection for checked parameter sampling, and optional <i><value></i> is the value that must be in values array, extracted from data source.</p> <div data-bbox="916 394 1449 748" style="background-color: #e6f2e6; padding: 10px;"><p>Values from Dr.Web LookupD are not available for all variables. You cannot apply the condition <i><scanning></i> to all variables. In the description of each variable below you can read if it possible to use Dr.Web LookupD to check its value.</p></div>

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 ()* is false as array of variable values is not empty;
- *x not in ()* is 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

Actions are divided into *final resolutions*, which determine whether to permit passing an object and *actions that change the values of some variable*, which can be used when checking conditions in the subsequent rules.

Ultimate Resolutions

Resolution	Description (Meaning)
Common Resolutions	
Pass	Let the traffic pass (allow to create a connection, send an object to the recipient). The further rules (if there are any) are not used.
Block as <i><reason></i>	<p>Block the traffic (prohibit to create a connection or send an object to a recipient). The further rules (if there are any) are not used.</p> <p>The <i><reason></i> for blocking is recorded in the log. The same reason is displayed in the browser notification shown to the user.</p> <p>The reasons for <code>Block</code> are the following:</p> <ul style="list-style-type: none">• <code>BlackList</code>—the data is blocked because it has been blacklisted by the user;• <code>_match</code>—the block happens because a web resource or file containing threat belongs to a category that triggers rule executing (for conditions <code>*_category in ()</code>). The <code>_match</code> variable contains the list of blocked categories which matched.

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.

Changing Value of a Variable

To change the variable value, the following instruction is used:

```
SET <variable> = ([<value 1>[, <value 2>[, ...]])
```

If there are no values in brackets, the list of the variable values will be cleared:

```
SET <переменная> = ( )
```



If the variable has only one value, the brackets are not used:

```
SET <переменная> = <значение>
```

Variables used in the rules

The variables are not case-sensitive. If the name of the variable is compound, you can write it either with or without an underscore. Thus, the names `variable_name`, `VariableName`, and `variablename` are actually the actually the equivalent variations of the same name. In the table below the names of all variables are written with an underscore.

Common variables

Variable	Description	Can be used in	
		conditions	actions (SET)
<code>protocol</code>	<p>Network protocol used by the connection.</p> <p>The variable can take multiple values.</p> <p>Allowed values: HTTP, SMTP, IMAP, POP3.</p> <p>Usage Aspects</p> <ul style="list-style-type: none">• The variable value can be defined only if SSL/TLS is not used or it was allowed to unwrap SSL.• The only possible value of the variable in the rules for Dr.Web ICAPD is HTTP.• A set of values for checking the variable value is can be read from a text file. <p>Examples:</p> <pre>protocol in (HTTP, SMTP) protocol in (POP3) protocol in file("/etc/file")</pre>	Yes	No
<code>sni_host</code>	<p>SNI host (address) to which the connection is established via SSL/TLS.</p> <p>Usage Aspects</p> <ul style="list-style-type: none">• If SSL is not used, the condition is false as the value of the variable is not defined.• You cannot the variable in the rules for Dr.Web ICAPD (it does not process SSL, so the condition always evaluates to false).	Yes	No



Variable	Description	Can be used in	
		conditions	actions (SET)
	<ul style="list-style-type: none">• A set of values for checking a variable value can be read from a text file.• You can use this variable together with the <code>proc</code> variable (see below). Examples: <pre>sni_host not in ('vk.com', 'ya.ru') sni_host in "LinuxFirewall.BlackList" sni_host in file("/etc/file")</pre>		
<code>sni_category</code>	<p>The list of categories (AdultContent, and so on) 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.</p> <p>The variable can take multiple values.</p> <p>Usage Aspects</p> <ul style="list-style-type: none">• If SSL is not used, the value of a variable is not defined and the condition evaluates to false.• You cannot use the variable in the rules for Dr.Web ICAPD (it does not process SSL, for that reason the condition always evaluates to false).• In the rules for Dr.Web ICAPD, a condition with <code>not in</code> will be <i>true</i>, even if the host does not belong to any of the predetermined categories ("safe" host).• If databases of web resource categories are not installed, it is impossible to use the variable in the rules (attempts to check if a condition in the rule is true will lead to the error x112).• A set of values for checking the variable value is can be read from a text file. Examples: <pre>sni_category not in (AdultContent, Chats) sni_category in</pre>	Yes	No



Variable	Description	Can be used in	
		conditions	actions (SET)
	<pre>"LinuxFirewall.BlockCategory" sni_category in (FreeEmail) sni_category not in file("/etc/file")</pre>		
url	<p>URL requested by the client. Can be compared with the specified string or with a regular expression.</p> <p>Usage Aspects</p> <ul style="list-style-type: none">• You can use this variable only in rules for Dr.Web ICAPD.• You can use Dr.Web LookupD to check the value of this variable.• A set of values for checking the value of the variable can be read from the file.• You can use this variable together with the <code>proc</code> variable (see below). <p>Examples:</p> <pre>url match ("drweb.com", "example\..*", "aaa.ru/") url match "ICAPD.Adlist" url not match LDAP@BadURLs url match file("/etc/file")</pre>	Yes	No
url_host	<p>URL/host to which the connection is established.</p> <p>Usage Aspects</p> <ul style="list-style-type: none">• You can set a value for this variable only if SSL/TLS is not used for connection or if SSL unwrapping is allowed.• You can use Dr.Web LookupD can to check the value of this variable.• A set of values for checking the variable value is can be read from a text file. <p>Examples:</p> <pre>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")</pre>	Yes	No



Variable	Description	Can be used in	
		conditions	actions (SET)
<code>url_category</code>	<p>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.</p> <p>The variable can take multiple values.</p> <p>Usage Aspects</p> <ul style="list-style-type: none">You can set a value for this variable only if SSL/TLS is not used for connection or if SSL unwrapping is allowed.In the rules Dr.Web ICAPD, a condition with <code>not in</code> will be <i>true</i>, even if the URL/host does not belong to any of the predetermined categories ("safe" URL/host). In the rules for Dr.Web Firewall for Linux (SpIDer Gate), the condition in this case will be <i>false</i>.If databases of web resource categories are not installed, it is impossible to use the variable in the 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 can be read from a text file. <p>Examples:</p> <pre>url_category not in (AdultContent, Chats) url_category in "LinuxFirewall.BlockCategory" url_category in (FreeEmail) url_category in file("/etc/file")</pre>	Yes	No
<code>threat_category</code>	<p>The list of categories the threat belongs to, which is found in the transferred data (according to information from virus databases).</p> <p>The variable can simultaneously contain a set of values.</p> <p>Usage Aspects</p> <ul style="list-style-type: none">You can set a value for this variable only if SSL/TLS is not used for connection or if SSL unwrapping is allowed.	Yes	No



Variable	Description	Can be used in	
		conditions	actions (SET)
	<ul style="list-style-type: none"> In the rules for Dr.Web ICAPD, a condition with <code>not in</code> will be <i>true</i>, even if the object does not contain threats from any of the predetermined categories ("safe" object). In the rules for Dr.Web Firewall for Linux (SpIDer Gate), the condition in this case will be <i>false</i>. A set of values for checking a variable value can be read from a text file. <p>Examples:</p> <pre>threat_category in "LinuxFirewall.BlockThreat" threat_category not in (Joke) threat_category in file("/etc/file")</pre>		
<code>user</code>	<p>The user on whose behalf the process sending (or receiving) the traffic has been launched.</p> <p>Usage Aspects</p> <ul style="list-style-type: none"> In the rules for Dr.Web ICAPD the name of the user who has authenticated on the proxy server is set as the value of the variable. If the proxy server does not support user authentication, the value of the variable is empty. You can use Dr.Web LookupD 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: <i><type of the source for LookupD>@<source of groups>@<required group></i>. Requests to Active Directory (AD@) return only lists of groups, therefore for these requests it is mandatory to use the <i>@<required group></i> part. A set of values for checking a variable value can be read from a text file. 	Yes	No



Variable	Description	Can be used in	
		conditions	actions (SET)
	<p>Examples:</p> <pre>user in ('user1', 'user2') user in AD@Winusergroups@Admins user in LDAP@AllowedUsers user not in file("/etc/file")</pre>		
src_ip	<p>The IP address of a host establishing the connection.</p> <p>Usage Aspects</p> <ul style="list-style-type: none">• You can use Dr.Web LookupD to check the value of this variable.• A set of values for checking a variable value can be read from a text file. <p>Examples:</p> <pre>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")</pre>	Yes	No
proc	<p>The process establishing the connection (the full executable path).</p> <p>Usage Aspects</p> <ul style="list-style-type: none">• You cannot use the variable in the rules for Dr.Web ICAPD(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 can be read from a text file.• You can use this variable together with <code>sni_host</code>, <code>url</code>, and <code>dst_address</code> (see below). <p>Examples:</p> <pre>proc in ('/usr/bin/ls') proc not in ('/home/user/myapp', '/bin/bin1') proc in</pre>	Yes	No



Variable	Description	Can be used in	
		conditions	actions (SET)
	<pre>"LinuxFirewall.ExcludedProc" proc in file("/etc/file")</pre>		
direction	<p>The type of message sent via the connection.</p> <p>Allowed values: <code>request</code> (client request), <code>response</code> (server reply).</p> <p>This variable cannot take multiple values; conditions of the <code>match</code> and <code>in type</code> cannot be applied.</p> <p>Examples:</p> <pre>direction request direction not response</pre>	Yes	No
divert	<p>The direction of the connection.</p> <p>Allowed values: <code>input</code> (incoming—created/initiated from outside the local host), <code>output</code> (outgoing—created/initiated on the local host).</p> <p>This variable cannot take multiple values; conditions of the <code>match</code> and <code>in type</code> cannot be applied.</p> <p>Examples:</p> <pre>divert input divert not output</pre>	Yes	No
content_type	<p>MIME type of data transferred via connection.</p> <p>Usage Aspects</p> <ul style="list-style-type: none">• You can set a value for this variable only if SSL/TLS is not used for connection or if SSL unwrapping is allowed.• The expression <code>"*/"</code> matches data of any MIME type and HTTP replies without the header <code>Content-Type</code>.• You can use Dr.Web LookupD can be used to check the value of this variable.• A set of values for checking a variable value can be read from a file.	Yes	No



Variable	Description	Can be used in	
		conditions	actions (SET)
	<p>Examples:</p> <pre>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")</pre>		
(<i>proc</i> , < <i>variable</i> >)	<p>Network activity of the process, where <i>proc</i> is the full process executable path (see above), and <<i>variable</i>> defines type of the activity and can be one of the following values:</p> <ul style="list-style-type: none"> • <i>sni_host</i>—SNI host (address), with which the connection is established via SSL/TLS; • <i>url</i>—URL requested by a client (see above); • <i>dst_address</i>—network address (<<i>IP address</i>>:<<i>port</i>>), with which the process establishes the connection. <p>Usage Aspects</p> <ul style="list-style-type: none"> • Using only with the condition <code>match ({<<i>Proc_reg</i>>, <<i>Var_reg</i>> [, ...]}, where <<i>Proc_reg</i>> is a regular expression for <i>proc</i>, and <<i>Var_reg</i>> is a regular expression for <<i>variable</i>>.</code> • You cannot use the variable in the rules for Dr.Web ICAPD (the component does not contain information about processes, for that reason the condition always evaluates to false). <p>Examples:</p> <pre>(proc, url) match ({"usr/bin/wget", "www\.\ya\.*"}) (proc, dst_address) match ({"usr/bin/.*", "192\.\168\.\1\.\d+:12345"})</pre>	Yes	No



Variable	Description	Can be used in	
		conditions	actions (SET)
<code>unwrap_ssl</code>	<p>Indicates whether the traffic transferred via SSL/TLS is unwrapped.</p> <p>Allowed values: <code>true</code>, <code>false</code>.</p> <p>Usage Aspects</p> <ul style="list-style-type: none">• The value of the variable cannot be empty. The instruction <code>SET unwrap_ssl = ()</code> is impossible.• You cannot use this variable in the condition. It is necessary only to control SSL unwrapping (for example, to display a webpage containing notification about blocking triggered by our side).• You cannot use the variable in the rules for Dr.Web ICAPD (it does not process SSL, changing of the variable does not influence rule processing). <p>Examples:</p> <pre>SET unwrap_ssl = TRUE set Unwrap_SSL = false</pre>	No	Yes
<code>http_templates_dir</code>	<p>The path to the directory where the block pages templates are stored.</p> <p>If a path starts with a / (forward slash), it is an absolute path; if it starts with any other symbol, it is a relative path. In the latter case it is given relative to the directory specified in the <code>TemplatesDir</code> parameter.</p> <p>Usage Aspects</p> <ul style="list-style-type: none">• Can be used for the HTTP(S) protocol. <p>Examples:</p> <pre>SET http_templates_dir = "/etc/mytemplates" set http_templates_dir = "templates_for_my_site"</pre>	No	Yes

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 that contain pornographic or erotic materials, dating sites, and so on
Violence	Websites that encourage violence or contain materials about various fatal accidents, and so on
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, and so on
Drugs	Websites that promote use, production or distribution of drugs, and so on
ObsceneLanguage	Websites that contain the obscene language (in titles, articles, and so on)
Chats	Websites that offer a real-time transmission of text messages
Terrorism	Websites that contain aggressive and propaganda materials or terroristic attacks descriptions, and so on.
FreeEmail	Websites that offer the possibility of free registration of an email box
SocialNetworks	Different social networking services: general, professional, corporate, interest-based; thematic dating websites
DueToCopyrightNotice	Websites, links to which are defined by the copyright holders of some copyrighted work (movies, music, and so on)
OnlineGames	Websites that provide access to games using the permanent internet connection
Anonymizers	Websites that allow the user to hide personal information and providing the access to the blocked web resources
CryptocurrencyMiningPool	Websites that provide an access to common services for cryptocurrencies mining
Jobs	Job search websites

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 used in rule conditions

Parameters, used in the component rules of Dr.Web Firewall for Linux (indicated with the `LinuxFirewall.` prefix):

Parameter	Description and Usage Example
Whitelist	White list, i.e. the list of domains the access to which is allowed, even if these domains are included in the database of categories. Examples: <pre>sni_host in "LinuxFirewall.Whitelist" : Pass url_host not in "LinuxFirewall.Whitelist" : Block as _match</pre>
Blacklist	Black list, i.e. the list of domains the access to which is blocked by the user (or the administrator). Examples: <pre>sni_host in "LinuxFirewall.Blacklist" : SET Unwrap_SSL = FALSE url_host in "LinuxFirewall.Blacklist" : Block as BlackList</pre>
BlockCategory	"Metaparameter": a list of names of web resource categories (Chats, AdultContent, and so on) for which the <code>Block*</code> parameters in the <code>[LinuxFirewall]</code> section are set to <code>Yes</code> .



Parameter	Description and Usage Example
	Examples: <pre>url_category in "LinuxFirewall.BlockCategory" : Block as _match sni_category in "LinuxFirewall.BlockCategory" : Block as BlackList</pre>
BlockThreat	"Metaparameter": a list of names of threat types (KnownVirus, Joke, and so on) for which the Block* parameters in the [LinuxFirewall] section are set to Yes. Examples: <pre>threat_category in "LinuxFirewall.BlockThreat" : Block as _match</pre>
ExcludedProc	The list of trusted processes, whose traffic must be excluded from scanning. Examples: <pre>proc in "LinuxFirewall.ExcludedProc" : Pass</pre>

Parameters, used in the component rules of Dr.Web ICAPD (indicated with the ICAPD. prefix):

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: <pre>url_host not in "ICAPD.Whitelist" : Block as BlackList</pre>
Blacklist	Black list contains the list of domains, the access to which is blocked by the user (or the administrator). Examples: <pre>url_host in "ICAPD.Blacklist" : Block as BlackList</pre>
Adlist	Advertisement list. Contains a list of regular expressions that describe advertising websites. The list is defined by the user (or administrator). Examples: <pre>url match "ICAPD.Adlist" : Block as BlackList</pre>
BlockCategory	"Metaparameter": a list of names of web resource categories (Chats, AdultContent, etc.) for which the Block* parameters in the [ICAPD] section are set to Yes. Examples: <pre>url_category in "ICAPD.BlockCategory" : Block as _match</pre>



Parameter	Description and Usage Example
BlockThreat	<p>“Metaparameter”: a list of names of threat types (KnownVirus, Joke, etc.) for which the Block* parameters in the [ICAPD] section are set to Yes.</p> <p>Examples:</p> <pre>threat_category in "ICAPD.BlockThreat" : Block as _match</pre>

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 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 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”, log the chain of rules 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 Dr.Web for UNIX Internet Gateways 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.

To Generate a Private SSL Key and a Certificate

1. To generate a private key (the RSA algorithm, the key length is 2048 bits), execute the command:

```
$ 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. To generate a certificate for the specified time period, based on the existing private key (in this case, for 365 days), execute the command:

```
$ 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 scan the contents of the generated certificate, use the command:

```
$ openssl x509 -noout -text -in certificate.crt
```

To Register a Certificate as a Trusted CA Certificate

1. Move or copy the certificate file to the system 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 directory containing certificates.

The example below performs all these three actions. This assumes that the current certificate directory is the trusted certificates directory `/etc/ssl/certs/` and the certificate that is registered as a trusted one is located in the `/home/user/ca.crt` file:

```
# cp /home/user/ca.crt ./
# ln -s ca.crt `openssl x509 -hash -noout -in ca.crt`.0
# c_rehash /etc/ssl/certs/
```

To create a signed certificate

1. Generate a request for signing a certificate (*Certificate Signing Request, CSR*) based on the existing private key. If the key is absent, generate it.

The request for signing is created with the 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 certificate, based on the request and the existing CA certificate, by using the command:

```
$ openssl x509 -req -days 365 -CA ca.crt -CAkey ca.key -set_serial 01 -in request.csr -out sigcert.crt
```



To create a signed certificate, you must have the following three files: the file of the root certificate `ca.crt` and its private key `ca.key` (the `certificate.crt` certificate and the `keyfile.key` key may be used instead of `ca.crt` and `ca.key`, then the obtained certificate will be self-signed), as well as 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
```

Repeat this procedure as many times as unique certificates you want to create. For example, every agent for distributed file scanning Dr.Web Network Checker within a scanning cluster should have its own key and certificate.



Modifying a signed certificate

Some browsers or mail clients may require modification of the signed certificate, used for authorization, to the PKCS12 format.

You can modify the certificate using the command:

```
# openssl pkcs12 -export -in sigcert.crt -out sigcert.pfx -inkey keyfile.key
```

`sigcert.crt` here is an existing file of the signed certificate. `keyfile.key` is a file of the corresponding private key. The modified certificate is saved to `sigcert.pfx`.



Appendix F. Known Errors

In this appendix

- [Recommendations for Error Identification](#)
- [Error Codes](#)
- [Errors without Code](#)



If you encounter an error that is not described in this section, please contact [technical support](#). Be ready to name the error code and describe steps to reproduce the issue.

Recommendations for Error Identification

- To identify the possible cause and the circumstances of the error, refer to the Dr.Web for UNIX Internet Gateways log (by default, it is located in the `/var/log/syslog` file or the `/var/log/messages` file, depending on OS). Also, you can use the following [command](#):

```
# drweb-ctl log
```

- 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, run 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 commands:

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

Errors Codes

Error message: *Error on monitor channel*

Error code: x1

Internal designation: EC_MONITOR_IPC_ERROR

Description: One or several components cannot connect with the [Dr.Web ConfigD](#) configuration daemon.

To identify the possible cause and the circumstances of the error, refer to the Dr.Web for UNIX Internet Gateways log (by default, it is located in the `/var/log/syslog` file or the `/var/log/messages` file, depending on OS). Also, you can use the [command](#) `drweb-ctl log`.



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 Dr.Web for UNIX Internet Gateways settings to the defaults.

To do this, 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 Dr.Web for UNIX Internet Gateways or its components, refer to sections [Installing Dr.Web for UNIX Internet Gateways](#) and [Uninstalling Dr.Web for UNIX Internet Gateways](#).

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

Internal designation: EC_ALREADY_IN_PROGRESS

Description: The operation is already in progress.

To identify the possible cause and the circumstances of the error, refer to the Dr.Web for UNIX Internet Gateways log (by default, it is located in the `/var/log/syslog` file or the `/var/log/messages` file, depending on OS). Also, you can use the [command](#) `drweb-ctl log`.

Resolving the error

1. Wait until operation is finished. If necessary, repeat the required action later.

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

Internal designation: EC_IN_PENDING_STATE

Description: The requested operation is in pending state.

To identify the possible cause and the circumstances of the error, refer to the Dr.Web for UNIX Internet



Gateways log (by default, it is located in the `/var/log/syslog` file or the `/var/log/messages` file, depending on OS). Also, you can use the [command](#) `drweb-ctl log`.

Resolving the error

1. Wait for the operation to start. If necessary, repeat the required action later.

If the error persists, contact [technical support](#) and be ready to name the error code.

Error message: *Interrupted by user*

Error code: `x4`

Internal designation: `EC_INTERRUPTED_BY_USER`

Description: The action has been terminated by the user.

To identify the possible cause and the circumstances of the error, refer to the Dr.Web for UNIX Internet Gateways log (by default, it is located in the `/var/log/syslog` file or the `/var/log/messages` file, depending on OS). Also, you can use the [command](#) `drweb-ctl log`.

Resolving the error

1. Repeat the required action later.

If the error persists, contact [technical support](#) and be ready to name the error code.

Error message: *Operation canceled*

Error code: `x5`

Internal designation: `EC_CANCELED`

Description: The action has been canceled.

To identify the possible cause and the circumstances of the error, refer to the Dr.Web for UNIX Internet Gateways log (by default, it is located in the `/var/log/syslog` file or the `/var/log/messages` file, depending on OS). Also, you can use the [command](#) `drweb-ctl log`.

Resolving the error

1. Repeat the required action later.

If the error persists, contact [technical support](#) and be ready to name the error code.

Error message: *IPC connection terminated*

Error code: `x6`

Internal designation: `EC_LINK_DISCONNECTED`

Description: An inter-process communication (IPC) connection with one of the Dr.Web for UNIX Internet Gateways components has terminated. It must have shut down after being idle for a long time.



To identify the possible cause and the circumstances of the error, refer to the Dr.Web for UNIX Internet Gateways log (by default, it is located in the `/var/log/syslog` file or the `/var/log/messages` file, depending on OS). Also, you can use the [command](#) `drweb-ctl log`.

Resolving the error

1. If the operation is not finished, repeat it later. 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`

Internal designation: `EC_BAD_MESSAGE_SIZE`

Description: A message of invalid size has been received during component inter-process communication (IPC).

To identify the possible cause and the circumstances of the error, refer to the Dr.Web for UNIX Internet Gateways log (by default, it is located in the `/var/log/syslog` file or the `/var/log/messages` file, depending on OS). Also, you can use the [command](#) `drweb-ctl log`.

Resolving the error

1. Reload Dr.Web for UNIX Internet Gateways by entering the 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`

Internal designation: `EC_BAD_MESSAGE_FORMAT`

Description: A message of invalid format has been received during component inter-process communication (IPC).

To identify the possible cause and the circumstances of the error, refer to the Dr.Web for UNIX Internet Gateways log (by default, it is located in the `/var/log/syslog` file or the `/var/log/messages` file, depending on OS). Also, you can use the [command](#) `drweb-ctl log`.

Resolving the error

1. Reload Dr.Web for UNIX Internet Gateways by entering the 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

Internal designation: EC_NOT_READY

Description: The required action cannot be performed because the necessary component or device has not been initialized yet.

To identify the possible cause and the circumstances of the error, refer to the Dr.Web for UNIX Internet Gateways log (by default, it is located in the `/var/log/syslog` file or the `/var/log/messages` file, depending on OS). Also, you can use the [command](#) `drweb-ctl log`.

Resolving the error

1. Repeat the required action later.

If the error persists, contact [technical support](#) and be ready to name the error code.

Error message: *Component is not installed*

Error code: x10

Internal designation: EC_NOT_INSTALLED

Description: The required operation cannot be performed because the necessary component has not been installed yet.

To identify the possible cause and the circumstances of the error, refer to the Dr.Web for UNIX Internet Gateways log (by default, it is located in the `/var/log/syslog` file or the `/var/log/messages` file, depending on OS). Also, you can use the [command](#) `drweb-ctl log`.

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 the installation or reinstallation of the necessary component does not help, reinstall Dr.Web for UNIX Internet Gateways.

For details on how to install and uninstall Dr.Web for UNIX Internet Gateways or its components, refer to sections [Installing Dr.Web for UNIX Internet Gateways](#) and [Uninstalling Dr.Web for UNIX Internet Gateways](#).

If the error persists, contact [technical support](#) and be ready to name the error code.

Error message: *Unexpected IPC message*

Error code: x11

Internal designation: EC_UNEXPECTED_MESSAGE

Description: An unexpected message is received during component inter-process communication (IPC).

To identify the possible cause and the circumstances of the error, refer to the Dr.Web for UNIX Internet Gateways log (by default, it is located in the `/var/log/syslog` file or the `/var/log/messages` file, depending on OS). Also, you can use the [command](#) `drweb-ctl log`.

**Resolving the error**

1. Reload Dr.Web for UNIX Internet Gateways by entering the 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

Internal designation: EC_PROTOCOL_VIOLATION

Description: Protocol violation has occurred during component inter-process communication (IPC).

To identify the possible cause and the circumstances of the error, refer to the Dr.Web for UNIX Internet Gateways log (by default, it is located in the `/var/log/syslog` file or the `/var/log/messages` file, depending on OS). Also, you can use the [command](#) `drweb-ctl log`.

Resolving the error

1. Reload Dr.Web for UNIX Internet Gateways by entering the 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

Internal designation: EC_UNKNOWN_STATE

Description: The required operation cannot be performed because one or more subsystems of Dr.Web for UNIX Internet Gateways are in unknown state.

To identify the possible cause and the circumstances of the error, refer to the Dr.Web for UNIX Internet Gateways log (by default, it is located in the `/var/log/syslog` file or the `/var/log/messages` file, depending on OS). Also, you can use the [command](#) `drweb-ctl log`.

Resolving the error

1. Repeat the operation.
2. If the error persists, restart Dr.Web for UNIX Internet Gateways 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

Internal designation: EC_NOT_A_DIRECTORY

Description: A relative path to a file or directory is specified whereas an absolute path is required.

To identify the possible cause and the circumstances of the error, refer to the Dr.Web for UNIX Internet Gateways log (by default, it is located in the `/var/log/syslog` file or the `/var/log/messages` file, depending on OS). Also, you can use the [command](#) `drweb-ctl log`.

Resolving the error

1. Change the path to the file or 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

Internal designation: EC_NO_MEMORY

Description: There is not enough memory to complete the requested operation.

To identify the possible cause and the circumstances of the error, refer to the Dr.Web for UNIX Internet Gateways log (by default, it is located in the `/var/log/syslog` file or the `/var/log/messages` file, depending on OS). Also, you can use the [command](#) `drweb-ctl log`.

Resolving the error

1. Increase size of available memory for Dr.Web for UNIX Internet Gateways processes (for example, by changing the limits with the `ulimit` command), restart Dr.Web for UNIX Internet Gateways and repeat the operation.



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`.

Restart Dr.Web for UNIX Internet Gateways by entering the 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

Internal designation: EC_IO_ERROR

Description: An input/output (I/O) error has occurred.

To identify the possible cause and the circumstances of the error, refer to the Dr.Web for UNIX Internet Gateways log (by default, it is located in the `/var/log/syslog` file or the `/var/log/messages` file, depending on OS). Also, you can use the [command](#) `drweb-ctl log`.

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

Internal designation: EC_NO_SUCH_ENTRY

Description: The specified object of the file system (file or directory) is missing. It may have been removed.

To identify the possible cause and the circumstances of the error, refer to the Dr.Web for UNIX Internet Gateways log (by default, it is located in the `/var/log/syslog` file or the `/var/log/messages` file, depending on OS). Also, you can use the [command](#) `drweb-ctl log`.

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

Internal designation: EC_PERMISSION_DENIED

Description: The component cannot access the specified file or directory. It may not have the required permission to access the item.

To identify the possible cause and the circumstances of the error, refer to the Dr.Web for UNIX Internet Gateways log (by default, it is located in the `/var/log/syslog` file or the `/var/log/messages` file, depending on OS). Also, you can use the [command](#) `drweb-ctl log`.

Resolving the error

1. Check whether the path is correct and whether the component has the required permissions. If it is necessary to access the object, change access permissions or elevate component 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

Internal designation: EC_NOT_A_DIRECTORY

Description: The specified object of the file system is not a directory.

To identify the possible cause and the circumstances of the error, refer to the Dr.Web for UNIX Internet Gateways log (by default, it is located in the `/var/log/syslog` file or the `/var/log/messages` file, depending on OS). Also, you can use the [command](#) `drweb-ctl log`.

Resolving the error

1. Check the path. Specify the correct path 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

Internal designation: EC_NOT_A_DIRECTORY

Description: The requested data is corrupted.

To identify the possible cause and the circumstances of the error, refer to the Dr.Web for UNIX Internet Gateways log (by default, it is located in the `/var/log/syslog` file or the `/var/log/messages` file, depending on OS). Also, you can use the [command](#) `drweb-ctl log`.

Resolving the error

1. Repeat the operation.
2. If the error persists, restart Dr.Web for UNIX Internet Gateways 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

Internal designation: EC_FILE_EXISTS

Description: There is already the file with the same name in the specified location.

To identify the possible cause and the circumstances of the error, refer to the Dr.Web for UNIX Internet Gateways log (by default, it is located in the `/var/log/syslog` file or the `/var/log/messages` file, depending on OS). Also, you can use the [command](#) `drweb-ctl log`.

**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

Internal designation: EC_READ_ONLY_FS

Description: The file system object (a file, a directory or a socket) that you are trying to modify is read-only.

To identify the possible cause and the circumstances of the error, refer to the Dr.Web for UNIX Internet Gateways log (by default, it is located in the `/var/log/syslog` file or the `/var/log/messages` file, depending on OS). Also, you can use the [command](#) `drweb-ctl log`.

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

Internal designation: EC_NETWORK_ERROR

Description: A network error has occurred.

To identify the possible cause and the circumstances of the error, refer to the Dr.Web for UNIX Internet Gateways log (by default, it is located in the `/var/log/syslog` file or the `/var/log/messages` file, depending on OS). Also, you can use the [command](#) `drweb-ctl log`.

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

Internal designation: EC_NOT_A_DRIVE

Description: The I/O device being accessed is not a drive.

To identify the possible cause and the circumstances of the error, refer to the Dr.Web for UNIX Internet



Gateways log (by default, it is located in the `/var/log/syslog` file or the `/var/log/messages` file, depending on OS). Also, you can use the [command](#) `drweb-ctl log`.

Resolving the error

1. Check the device name. Change the path so that it indicate 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`

Internal designation: `EC_UNEXPECTED_EOF`

Description: The end of file has been reached unexpectedly when reading data.

To identify the possible cause and the circumstances of the error, refer to the Dr.Web for UNIX Internet Gateways log (by default, it is located in the `/var/log/syslog` file or the `/var/log/messages` file, depending on OS). Also, you can use the [command](#) `drweb-ctl log`.

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`

Internal designation: `EC_FILE_WAS_CHANGED`

Description: The file being scanned has been modified.

To identify the possible cause and the circumstances of the error, refer to the Dr.Web for UNIX Internet Gateways log (by default, it is located in the `/var/log/syslog` file or the `/var/log/messages` file, depending on OS). Also, you can use the [command](#) `drweb-ctl log`.

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`

Internal designation: `EC_NOT_A_REGULAR_FILE`

Description: The object being accessed is not a regular file (it may a directory, a socket, etc.).



To identify the possible cause and the circumstances of the error, refer to the Dr.Web for UNIX Internet Gateways log (by default, it is located in the `/var/log/syslog` file or the `/var/log/messages` file, depending on OS). Also, you can use the [command](#) `drweb-ctl log`.

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`

Internal designation: `EC_NAME_ALREADY_IN_USE`

Description: The object cannot be created as there is an object with the same name in the file system.

To identify the possible cause and the circumstances of the error, refer to the Dr.Web for UNIX Internet Gateways log (by default, it is located in the `/var/log/syslog` file or the `/var/log/messages` file, depending on OS). Also, you can use the [command](#) `drweb-ctl log`.

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`

Internal designation: `EC_HOST_OFFLINE`

Description: A remote host cannot be accessed via the network.

To identify the possible cause and the circumstances of the error, refer to the Dr.Web for UNIX Internet Gateways log (by default, it is located in the `/var/log/syslog` file or the `/var/log/messages` file, depending on OS). Also, you can use the [command](#) `drweb-ctl log`.

Resolving the error

1. Check whether the required host is available. If necessary, change the host 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`

Internal designation: `EC_LIMIT_REACHED`



Description: The limit defined for the use of a certain resource has been reached.

To identify the possible cause and the circumstances of the error, refer to the Dr.Web for UNIX Internet Gateways log (by default, it is located in the `/var/log/syslog` file or the `/var/log/messages` file, depending on OS). Also, you can use the [command](#) `drweb-ctl log`.

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`

Internal designation: `EC_CROSS_DEVICE_LINK`

Description: The file cannot be restored as restoring implies moving it between two different mountain points.

To identify the possible cause and the circumstances of the error, refer to the Dr.Web for UNIX Internet Gateways log (by default, it is located in the `/var/log/syslog` file or the `/var/log/messages` file, depending on OS). Also, you can use the [command](#) `drweb-ctl log`.

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`

Internal designation: `EC_UNPACKING_ERROR`

Description: Archive unpacking failed.

To identify the possible cause and the circumstances of the error, refer to the Dr.Web for UNIX Internet Gateways log (by default, it is located in the `/var/log/syslog` file or the `/var/log/messages` file, depending on OS). Also, you can use the [command](#) `drweb-ctl log`.

Resolving the error

1. Make sure that the archive 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

Internal designation: EC_BASE_CORRUPTED

Description: Virus databases are corrupted.

To identify the possible cause and the circumstances of the error, refer to the Dr.Web for UNIX Internet Gateways log (by default, it is located in the `/var/log/syslog` file or the `/var/log/messages` file, depending on OS). Also, you can use the [command](#) `drweb-ctl log`.

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 **General Settings** page of the [web interface](#), if installed.
 - You also can 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 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

Internal designation: EC_OLD_BASE_VERSION

Description: Current virus databases are designed for an earlier program version.

To identify the possible cause and the circumstances of the error, refer to the Dr.Web for UNIX Internet Gateways log (by default, it is located in the `/var/log/syslog` file or the `/var/log/messages` file, depending on OS). Also, you can use the [command](#) `drweb-ctl log`.

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 **General Settings** page of the [web interface](#), if installed.



- You also can 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 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

Internal designation: EC_EMPTY_BASE

Description: Virus databases are empty.

To identify the possible cause and the circumstances of the error, refer to the Dr.Web for UNIX Internet Gateways log (by default, it is located in the `/var/log/syslog` file or the `/var/log/messages` file, depending on OS). Also, you can use the [command](#) `drweb-ctl log`.

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 **General Settings** page of the [web interface](#), if installed.
- You also can 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 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

Internal designation: EC_CAN_NOT_BE_CURED

Description: The **Cure** action has been applied to an incurable object.

To identify the possible cause and the circumstances of the error, refer to the Dr.Web for UNIX Internet Gateways log (by default, it is located in the `/var/log/syslog` file or the `/var/log/messages` file, depending on OS). Also, you can use the [command](#) `drweb-ctl log`.

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

Internal designation: EC_INVALID_BASE_SET

Description: The current combination of virus databases cannot be supported.

To identify the possible cause and the circumstances of the error, refer to the Dr.Web for UNIX Internet Gateways log (by default, it is located in the `/var/log/syslog` file or the `/var/log/messages` file, depending on OS). Also, you can use the [command](#) `drweb-ctl log`.

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 **General Settings** page of the [web interface](#), if installed.
 - You also can 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 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

Internal designation: EC_SCAN_LIMIT_REACHED

Description: The specified limits have been exceeded during the scanning of the object.

To identify the possible cause and the circumstances of the error, refer to the Dr.Web for UNIX Internet Gateways log (by default, it is located in the `/var/log/syslog` file or the `/var/log/messages` file, depending on OS). Also, you can use the [command](#) `drweb-ctl log`.

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

Internal designation: EC_AUTH_FAILED

Description: Invalid user credentials have been used for authentication.

To identify the possible cause and the circumstances of the error, refer to the Dr.Web for UNIX Internet Gateways log (by default, it is located in the `/var/log/syslog` file or the `/var/log/messages` file, depending on OS). Also, you can use the [command](#) `drweb-ctl log`.

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

Internal designation: EC_NOT_AUTHORIZED

Description: The user does not have enough rights to perform the requested operation.

To identify the possible cause and the circumstances of the error, refer to the Dr.Web for UNIX Internet Gateways log (by default, it is located in the `/var/log/syslog` file or the `/var/log/messages` file, depending on OS). Also, you can use the [command](#) `drweb-ctl log`.

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

Internal designation: EC_INVALID_TOKEN

Description: One of Dr.Web for UNIX Internet Gateways components provides invalid authorization token on attempt to access the operation, requiring elevated privileges.

To identify the possible cause and the circumstances of the error, refer to the Dr.Web for UNIX Internet Gateways log (by default, it is located in the `/var/log/syslog` file or the `/var/log/messages` file, depending on OS). Also, you can use the [command](#) `drweb-ctl log`.

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: *DBMS General Error*

Error code: x50

Internal designation: EC_DB_COMMON_ERROR

Description: The request to DBMS Server made by Dr.Web LookupD was not successful.

To identify the possible cause and the circumstances of the error, refer to the Dr.Web for UNIX Internet Gateways log (by default, it is located in the `/var/log/syslog` file or the `/var/log/messages` file, depending on OS). Also, you can use the [command](#) `drweb-ctl log`.

Refer to the DMBS server log as well.

If the error persists, contact [technical support](#) and be ready to name the error code.

Error message: *Cannot open database*



Error code: x51

Internal designation: EC_DB_OPEN_ERROR

Description: The database to which the Dr.Web LookupD is trying to connect is unavailable.

To identify the possible cause and the circumstances of the error, refer to the Dr.Web for UNIX Internet Gateways log (by default, it is located in the `/var/log/syslog` file or the `/var/log/messages` file, depending on OS). Also, you can use the [command](#) `drweb-ctl log`.

Refer to the DMBS server log as well.

If the error persists, contact [technical support](#) and be ready to name the error code.

Error message: *Connection closed by DBMS*

Error code: x52

Internal designation: EC_DB_CONN_CLOSED

Description: The connection was closed by the DBMS server.

To identify the possible cause and the circumstances of the error, refer to the Dr.Web for UNIX Internet Gateways log (by default, it is located in the `/var/log/syslog` file or the `/var/log/messages` file, depending on OS). Also, you can use the [command](#) `drweb-ctl log`.

Refer to the DMBS server log as well.

If the error persists, contact [technical support](#) and be ready to name the error code.

Error message: *Invalid argument*

Error code: x60

Internal designation: EC_INVALID_ARGUMENT

Description: An invalid argument has been used when trying to execute a command.

To identify the possible cause and the circumstances of the error, refer to the Dr.Web for UNIX Internet Gateways log (by default, it is located in the `/var/log/syslog` file or the `/var/log/messages` file, depending on OS). Also, you can use the [command](#) `drweb-ctl log`.

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

Internal designation: EC_INVALID_OPERATION



Description: An attempt to run an invalid command has been detected.

To identify the possible cause and the circumstances of the error, refer to the Dr.Web for UNIX Internet Gateways log (by default, it is located in the `/var/log/syslog` file or the `/var/log/messages` file, depending on OS). Also, you can use the [command](#) `drweb-ctl log`.

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`

Internal designation: `EC_ROOT_ONLY`

Description: Superuser privileges are required to perform the action.

To identify the possible cause and the circumstances of the error, refer to the Dr.Web for UNIX Internet Gateways log (by default, it is located in the `/var/log/syslog` file or the `/var/log/messages` file, depending on OS). Also, you can use the [command](#) `drweb-ctl log`.

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 centralized protection mode*

Error code: `x63`

Internal designation: `EC_STANDALONE_MODE_ONLY`

Description: The required action can be performed in standalone [mode](#) only.

To identify the possible cause and the circumstances of the error, refer to the Dr.Web for UNIX Internet Gateways log (by default, it is located in the `/var/log/syslog` file or the `/var/log/messages` file, depending on OS). Also, you can use the [command](#) `drweb-ctl log`.

Resolving the error

1. Change Dr.Web for UNIX Internet Gateways operation mode to standalone mode and repeat the operation.
2. To switch Dr.Web for UNIX Internet Gateways to standalone mode:
 - clear the **Enable centralized protection mode** check box on the **Centralized protection** of the [web interface](#), 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

Internal designation: EC_NON_SUPPORTED_OS

Description: Dr.Web for UNIX Internet Gateways does not support the operating system installed on the host.

To identify the possible cause and the circumstances of the error, refer to the Dr.Web for UNIX Internet Gateways log (by default, it is located in the `/var/log/syslog` file or the `/var/log/messages` file, depending on OS). Also, you can use the [command](#) `drweb-ctl log`.

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

Internal designation: EC_NOT_IMPLEMENTED

Description: The required features of one or more components are not implemented in the current version.

To identify the possible cause and the circumstances of the error, refer to the Dr.Web for UNIX Internet Gateways log (by default, it is located in the `/var/log/syslog` file or the `/var/log/messages` file, depending on OS). Also, you can use the [command](#) `drweb-ctl log`.

Resolving the error

1. Restore Dr.Web for UNIX Internet Gateways 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 Dr.Web for UNIX Internet Gateways 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

Internal designation: EC_UNKNOWN_SECTION

Description: The [configuration file](#) contains parameters unknown or non-supported in the current version of Dr.Web for UNIX Internet Gateways.

To identify the possible cause and the circumstances of the error, refer to the Dr.Web for UNIX Internet Gateways log (by default, it is located in the `/var/log/syslog` file or the `/var/log/messages` file, depending on OS). Also, you can use the [command](#) `drweb-ctl log`.

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 Dr.Web for UNIX Internet Gateways 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

Internal designation: EC_UNKNOWN_SECTION

Description: The [configuration file](#) contains sections unknown or non-supported in the current version of Dr.Web for UNIX Internet Gateways.

To identify the possible cause and the circumstances of the error, refer to the Dr.Web for UNIX Internet Gateways log (by default, it is located in the `/var/log/syslog` file or the `/var/log/messages` file, depending on OS). Also, you can use the [command](#) `drweb-ctl log`.

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 Dr.Web for UNIX Internet Gateways 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

Internal designation: EC_INVALID_OPTION_VALUE

Description: One or more parameters in the [configuration file](#) have invalid values.

To identify the possible cause and the circumstances of the error, refer to the Dr.Web for UNIX Internet Gateways log (by default, it is located in the `/var/log/syslog` file or the `/var/log/messages` file, depending on OS). Also, you can use the [command](#) `drweb-ctl log`.

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 can also restore parameter value to the default.

2. You can 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 did not help, restore Dr.Web for UNIX Internet Gateways 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

Internal designation: EC_INVALID_STATE

Description: Dr.Web for UNIX Internet Gateways or one of the components is in invalid state to complete the required operation.



To identify the possible cause and the circumstances of the error, refer to the Dr.Web for UNIX Internet Gateways log (by default, it is located in the `/var/log/syslog` file or the `/var/log/messages` file, depending on OS). Also, you can use the [command](#) `drweb-ctl log`.

Resolving the error

1. Repeat the required action later.
2. If the error persists, restart Dr.Web for UNIX Internet Gateways 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`

Internal designation: `EC_NOT_LIST_OPTION`

Description: In the [configuration file](#) a list of values is attributed to a single-valued parameter.

To identify the possible cause and the circumstances of the error, refer to the Dr.Web for UNIX Internet Gateways log (by default, it is located in the `/var/log/syslog` file or the `/var/log/messages` file, depending on OS). Also, you can use the [command](#) `drweb-ctl log`.

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 can also restore parameter value to the default.

2. You can 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 did not help, restore Dr.Web for UNIX Internet Gateways 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

Internal designation: EC_INVALID_TAG

Description: An incorrect or invalid (non-existent) tag has been used in the name of a data source Dr.Web LookupD component interacts with.

To identify the possible cause and the circumstances of the error, refer to the Dr.Web for UNIX Internet Gateways log (by default, it is located in the `/var/log/syslog` file or the `/var/log/messages` file, depending on OS). Also, you can use the [command](#) `drweb-ctl log`.

Resolving the error

1. Check the correctness of the tag spelling. If you find an error, edit the appropriate section in 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

Internal designation: EC_RECORD_NOT_FOUND

Description: The threat record is missing (it may have already been processed by another Dr.Web for UNIX Internet Gateways component).

To identify the possible cause and the circumstances of the error, refer to the Dr.Web for UNIX Internet Gateways log (by default, it is located in the `/var/log/syslog` file or the `/var/log/messages` file, depending on OS). Also, you can use the [command](#) `drweb-ctl log`.

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

Internal designation: EC_RECORD_BUSY

Description: The record is already being processed by another component.

To identify the possible cause and the circumstances of the error, refer to the Dr.Web for UNIX Internet Gateways log (by default, it is located in the `/var/log/syslog` file or the `/var/log/messages` file, depending on OS). Also, you can use the [command](#) `drweb-ctl log`.

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

Internal designation: EC_RECORD_BUSY

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 component processed the threat).

To identify the possible cause and the circumstances of the error, refer to the Dr.Web for UNIX Internet Gateways log (by default, it is located in the `/var/log/syslog` file or the `/var/log/messages` file, depending on OS). Also, you can use the [command](#) `drweb-ctl log`.

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: *Update zone is not provided it the cloud*

Error code: x83

Internal designation: EC_NO_ZONE_IN_CLOUD

Description: The attempt to update using Dr.Web Cloud turned unsuccessful.

To identify the possible cause and the circumstances of the error, refer to the Dr.Web for UNIX Internet Gateways log (by default, it is located in the `/var/log/syslog` file or the `/var/log/messages` file, depending on OS). Also, you can use the [command](#) `drweb-ctl log`.

Resolving the error

1. Repeat the required action later.

If the error persists, contact [technical support](#) and be ready to name the error code.

Error message: *Update zone is not provided on disk*

Error code: x84

Internal designation: EC_NO_ZONE_ON_DISK

Description: The attempt to update the virus bases in the offline mode turned unsuccessful.

To identify the possible cause and the circumstances of the error, refer to the Dr.Web for UNIX Internet Gateways log (by default, it is located in the `/var/log/syslog` file or the `/var/log/messages` file, depending on OS). Also, you can use the [command](#) `drweb-ctl log`.

Resolving the error

1. Make sure that the path to the device used for updating is correct.
2. Make sure that the user under which you try to update the bases has enough read permissions on the directory containing the updates.



If the error persists, contact [technical support](#) and be ready to name the error code.

Error message: *Cannot backup before update*

Error code: x89

Internal designation: EC_BACKUP_FAILED

Description: Prior to downloading the updates from the updates server, an attempt to make a backup copy of the files to be updated failed.

To identify the possible cause and the circumstances of the error, refer to the Dr.Web for UNIX Internet Gateways log (by default, it is located in the `/var/log/syslog` file or the `/var/log/messages` file, depending on OS). Also, you can use the [command](#) `drweb-ctl log`.

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 can use the [commands](#) of the command-line management tool.

To view current parameter value, execute the 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 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 properties.
4. If the error persists, reinstall the `drweb-update` package.

For details on how to install and uninstall Dr.Web for UNIX Internet Gateways or its components, refer to sections [Installing Dr.Web for UNIX Internet Gateways](#) and [Uninstalling Dr.Web for UNIX Internet Gateways](#).

If the error persists, contact [technical support](#) and be ready to name the error code.



Error message: *Invalid DRL file*

Error code: x90

Internal designation: EC_BAD_DRL_FILE

Description: The integrity of one of the files with the list of update servers has been violated.

To identify the possible cause and the circumstances of the error, refer to the Dr.Web for UNIX Internet Gateways log (by default, it is located in the `/var/log/syslog` file or the `/var/log/messages` file, depending on OS). Also, you can use the [command](#) `drweb-ctl log`.

Resolving the error

1. Check the path to the file with the list of servers and change the path if necessary (parameters with `*DrlDir` in [section](#) [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 can use the [commands](#) of the command-line management tool.

To view the current parameter value, use the command (`<*DrlDirPath>` needs to be substituted with a specified parameter name. If the parameter name is unclear, refer to all parameter values in the section, skipping the command part in square brackets):

```
$ drweb-ctl cfshow Update[.<*DrlDir>]
```

To set new parameter value, execute the command (`<*DrlDir>` needs to be substituted with a specified parameter name):

```
# drweb-ctl cfset Update.<*DrlDir> <new path>
```

To restore parameter value to the default, execute the command (`<*DrlDir>` needs to be substituted with a specified parameter name):

```
# drweb-ctl cfset Update.<*DrlDir> -r
```

2. Update virus databases:
 - click **Update** on the **Main** page of the [web interface](#), if 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 Dr.Web for UNIX Internet Gateways or its components, refer to sections [Installing Dr.Web for UNIX Internet Gateways](#) and [Uninstalling Dr.Web for UNIX Internet Gateways](#).

If the error persists, contact [technical support](#) and be ready to name the error code.

Error message: *Invalid LST file*

Error code: x91

Internal designation: EC_BAD_LST_FILE



Description: The integrity of the file with the list of updated virus databases has been violated.

To identify the possible cause and the circumstances of the error, refer to the Dr.Web for UNIX Internet Gateways log (by default, it is located in the `/var/log/syslog` file or the `/var/log/messages` file, depending on OS). Also, you can use the [command](#) `drweb-ctl log`.

Resolving the error

1. Update virus databases again after some time:
 - click **Update** on the **Main** page of the [web interface](#), if 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 Dr.Web for UNIX Internet Gateways or its components, refer to sections [Installing Dr.Web for UNIX Internet Gateways](#) and [Uninstalling Dr.Web for UNIX Internet Gateways](#).

If the error persists, contact [technical support](#) and be ready to name the error code.

Error message: *Invalid compressed file*

Error code: `x92`

Internal designation: `EC_BAD_LZMA_FILE`

Description: An integrity violation of the downloaded file containing updates has been detected.

To identify the possible cause and the circumstances of the error, refer to the Dr.Web for UNIX Internet Gateways log (by default, it is located in the `/var/log/syslog` file or the `/var/log/messages` file, depending on OS). Also, you can use the [command](#) `drweb-ctl log`.

Resolving the error

1. Update virus databases again after some time:
 - click **Update** on the **Main** page of the [web interface](#), if 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`

Internal designation: `EC_PROXY_AUTH_ERROR`

Description: The program fails to connect to update servers via the proxy server specified in the settings.



To identify the possible cause and the circumstances of the error, refer to the Dr.Web for UNIX Internet Gateways log (by default, it is located in the `/var/log/syslog` file or the `/var/log/messages` file, depending on OS). Also, you can use the [command](#) `drweb-ctl log`.

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 can use the [commands](#) of the command-line management tool.

To view current parameter value, execute the 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 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`

Internal designation: `EC_NO_UPDATE_SERVERS`

Description: The program fails to connect to any of the update servers.

To identify the possible cause and the circumstances of the error, refer to the Dr.Web for UNIX Internet Gateways log (by default, it is located in the `/var/log/syslog` file or the `/var/log/messages` file, depending on OS). Also, you can use the [command](#) `drweb-ctl log`.

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 can use the [commands](#) of the command-line management tool.

To view current parameter value, execute the 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 update servers used is displayed in parameters `*Dr1Dir` in [Update] section of configuration file.



If parameters `*CustomDr1Dir` 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 `*Dr1Dir`, is ignored).

- To view and set the connection parameters, go to the **Updater** page of the [web interface](#) (if it is installed).
- You also can use the [commands](#) of the command-line management tool.

To view the current parameter value, use the command (`<*Dr1DirPath>` needs to be substituted with a specified parameter name. If the parameter name is unclear, refer to all parameter values in the section, skipping the command part in square brackets):

```
$ drweb-ctl cfshow Update[.<*Dr1Dir>]
```

To set new parameter value, execute the command (`<*Dr1Dir>` needs to be substituted with a specified parameter name):

```
# drweb-ctl cfset Update.<*Dr1Dir> <new path>
```

To restore parameter value to the default, execute the command (`<*Dr1Dir>` needs to be substituted with a specified parameter name):

```
# drweb-ctl cfset Update.<*Dr1Dir> -r
```

4. Update virus databases:

- click **Update** on the **Main** page of the [web interface](#), if 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

Internal designation: EC_BAD_KEY_FORMAT

Description: The key file format is violated.

To identify the possible cause and the circumstances of the error, refer to the Dr.Web for UNIX Internet Gateways log (by default, it is located in the `/var/log/syslog` file or the `/var/log/messages` file, depending on OS). Also, you can use the [command](#) `drweb-ctl log`.

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 **General Settings** page of the [web interface](#), if installed.
 - You also can use the [commands](#) of the command-line management tool.

To view current parameter value, execute the 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 can use the license activation form at the bottom of the **Main** page of the [web interface](#), if installed.
4. You can view current license options on user 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

Internal designation: EC_EXPIRED_KEY

Description: The license has expired.

To identify the possible cause and the circumstances of the error, refer to the Dr.Web for UNIX Internet Gateways log (by default, it is located in the `/var/log/syslog` file or the `/var/log/messages` file, depending on OS). Also, you can use the [command](#) `drweb-ctl log`.



Resolving the error

1. Purchase a new license and install the 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 purchased key file, you can use the license activation form at the bottom of the **Main** page of the [web interface](#), if installed.
3. You can view current license options on user 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

Internal designation: EC_NETWORK_TIMEDOUT

Description: Network operation timed out (possibly, a remote host stops responding unexpectedly or the required connection fails).

To identify the possible cause and the circumstances of the error, refer to the Dr.Web for UNIX Internet Gateways log (by default, it is located in the `/var/log/syslog` file or the `/var/log/messages` file, depending on OS). Also, you can use the [command](#) `drweb-ctl log`.

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

Internal designation: EC_BAD_CHECKSUM

Description: The checksum of the downloaded file with updates is corrupted.

To identify the possible cause and the circumstances of the error, refer to the Dr.Web for UNIX Internet Gateways log (by default, it is located in the `/var/log/syslog` file or the `/var/log/messages` file, depending on OS). Also, you can use the [command](#) `drweb-ctl log`.

Resolving the error

1. Update virus databases again after some time:
 - click **Update** on the **Main** page of the [web interface](#), if 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 trial license*

Error code: x99

Internal designation: EC_BAD_TRIAL_KEY

Description: The demo key file is invalid (for example, it may have been received from another computer).

To identify the possible cause and the circumstances of the error, refer to the Dr.Web for UNIX Internet Gateways log (by default, it is located in the `/var/log/syslog` file or the `/var/log/messages` file, depending on OS). Also, you can use the [command](#) `drweb-ctl log`.

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 purchased key file, you can use the license activation form at the bottom of the **Main** page of the [web interface](#), if installed.
3. You can view current license options on user 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: *Blocked license key*

Error code: x100

Internal designation: EC_BLOCKED_LICENSE

Description: The license in use is blocked (the terms of use of Dr.Web for UNIX Internet Gateways may be violated).

To identify the possible cause and the circumstances of the error, refer to the Dr.Web for UNIX Internet Gateways log (by default, it is located in the `/var/log/syslog` file or the `/var/log/messages` file, depending on OS). Also, you can use the [command](#) `drweb-ctl log`.

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 can use the license activation form at the bottom of the **Main** page of the [web interface](#), if installed.
3. You can view current license options on user 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

Internal designation: EC_BAD_LICENSE

Description: The license is designed for other product or does not allow the operation of Dr.Web for UNIX Internet Gateways components.

To identify the possible cause and the circumstances of the error, refer to the Dr.Web for UNIX Internet Gateways log (by default, it is located in the `/var/log/syslog` file or the `/var/log/messages` file, depending on OS). Also, you can use the [command](#) `drweb-ctl log`.

Resolving the error

1. Purchase a new license and install the 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 can use the license activation form at the bottom of the **Main** page of the [web interface](#), if installed.
3. You can view current license options on the user 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

Internal designation: EC_BAD_CONFIG

Description: One or more Dr.Web for UNIX Internet Gateways components cannot operate because of incorrect configuration settings.

To identify the possible cause and the circumstances of the error, refer to the Dr.Web for UNIX Internet Gateways log (by default, it is located in the `/var/log/syslog` file or the `/var/log/messages` file, depending on OS). Also, you can use the [command](#) `drweb-ctl log`.

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 caused by Dr.Web Firewall for Linux, a conflict with another firewall is probably occurring. 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 Dr.Web for UNIX Internet Gateways by executing the command:

```
# service drweb-configd restart
```

or

```
# drweb-ctl reload
```



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 reported 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 did not help, restore Dr.Web for UNIX Internet Gateways 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 Dr.Web for UNIX Internet Gateways 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

Internal designation: EC_BAD_EXECUTABLE

Description: The executable file of the component is corrupted.

To identify the possible cause and the circumstances of the error, refer to the Dr.Web for UNIX Internet Gateways log (by default, it is located in the `/var/log/syslog` file or the `/var/log/messages` file, depending on OS). Also, you can use the [command](#) `drweb-ctl log`.

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. Check the executable path to the component in the Dr.Web for UNIX Internet Gateways 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 Dr.Web for UNIX Internet Gateways or its components, refer to sections [Installing Dr.Web for UNIX Internet Gateways](#) and [Uninstalling Dr.Web for UNIX Internet Gateways](#).

If the error persists, contact [technical support](#) and be ready to name the error code.

Error message: *Core engine is not available*

Error code: x105

Internal designation: EC_NO_CORE_ENGINE

Description: The file of Dr.Web Virus-Finding Engine is missing or unavailable (required for threat detection).

To identify the possible cause and the circumstances of the error, refer to the Dr.Web for UNIX Internet Gateways log (by default, it is located in the `/var/log/syslog` file or the `/var/log/messages` file, depending on OS). Also, you can use the [command](#) `drweb-ctl log`.

Resolving the error

1. Check the path to the `drweb32.dll` scan 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 **General Settings** page of the [web interface](#), if installed.
 - You also can 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 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 Dr.Web for UNIX Internet Gateways or its components, refer to sections [Installing Dr.Web for UNIX Internet Gateways](#) and [Uninstalling Dr.Web for UNIX Internet Gateways](#).

If the error persists, contact [technical support](#) and be ready to name the error code.

Error message: *No virus databases*

Error code: x106

Internal designation: EC_NO_VIRUS_BASES

Description: Virus databases have not been found.

To identify the possible cause and the circumstances of the error, refer to the Dr.Web for UNIX Internet Gateways log (by default, it is located in the `/var/log/syslog` file or the `/var/log/messages` file, depending on OS). Also, you can use the [command](#) `drweb-ctl log`.

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 **General Settings** page of the [web interface](#), if installed.
 - You also can 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 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

Internal designation: EC_APP_TERMINATED

Description: A component has shut down (e.g., because of the user command or being idle).



To identify the possible cause and the circumstances of the error, refer to the Dr.Web for UNIX Internet Gateways log (by default, it is located in the `/var/log/syslog` file or the `/var/log/messages` file, depending on OS). Also, you can use the [command](#) `drweb-ctl log`.

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 did not help, restore Dr.Web for UNIX Internet Gateways 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 Dr.Web for UNIX Internet Gateways 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`

Internal designation: `EC_APP_CRASHED`

Description: A component has shut down because of failure.

To identify the possible cause and the circumstances of the error, refer to the Dr.Web for UNIX Internet Gateways log (by default, it is located in the `/var/log/syslog` file or the `/var/log/messages` file, depending on OS). Also, you can use the [command](#) `drweb-ctl log`.

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 did not help, restore Dr.Web for UNIX Internet Gateways 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 Dr.Web for UNIX Internet Gateways after clearing the contents of the configuration file by executing the command:

```
# service drweb-configd restart
```

4. If the error persists after restoring Dr.Web for UNIX Internet Gateways settings, reinstall the component package.

For details on how to install and uninstall Dr.Web for UNIX Internet Gateways or its components, refer to sections [Installing Dr.Web for UNIX Internet Gateways](#) and [Uninstalling Dr.Web for UNIX Internet Gateways](#).

If the error persists, contact [technical support](#) and be ready to name the error code.

Error message: *Incompatible software detected*

Error code: x109

Internal designation: EC_INCOMPATIBLE

Description: One or several components of Dr.Web for UNIX Internet Gateways cannot operate properly. There is software that impedes their operation in your system.

To identify the possible cause and the circumstances of the error, refer to the Dr.Web for UNIX Internet Gateways log (by default, it is located in the `/var/log/syslog` file or the `/var/log/messages` file, depending on OS). Also, you can use the [command](#) `drweb-ctl log`.

Resolving the error

1. If this error is reported by SpIDer Gate, an incompatible program may be running in the operating system. This program may generate 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 configures the NetFilter system firewall may check the integrity of the specified rule system and rewrite it. This may be the main reason of SpIDer Gate conflict with such applications.

Reconfigure the incompatible software lest it interfere in SpIDer Gate operation. If it is not possible, disable the software lest it load at the operating system startup any more. You can try to configure the SuseFirewall2 application (in SUSE Linux OS), following the steps below:

- 1) open the configuration file of SuseFirewall2 (by default, this is the `/etc/sysconfig/SuSEfirewall2` file);
- 2) find the following lines:

```
# 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 `FW_LO_NOTRACK` parameter value to "no":

```
FW_LO_NOTRACK="no"
```

4) Restart SuseFirewall2:

```
# rcSuSEfirewall2 restart
```



If the `FW_LO_NOTRACK` option is missing in SuseFirewall2 settings, stop the application and disable its launch at system startup.

After reconfiguring or disabling the conflict application, restart SpIDer Gate.

2. If the error is reported 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: *No web resource databases*

Error code: x112

Internal designation: EC_NO_DWS_BASES

Description: Databases of web resource categories are missing.

To identify the possible cause and the circumstances of the error, refer to the Dr.Web for UNIX Internet Gateways log (by default, it is located in the `/var/log/syslog` file or the `/var/log/messages` file, depending on OS). Also, you can use the [command](#) `drweb-ctl log`.

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 **General Settings** page of the [web interface](#), if installed.
 - You also can 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 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: *MeshD is not available*

Error code: x114

Internal designation: EC_NO_MESH D

Description: The Dr.Web MeshD component required for load balancing during the scanning of files is missing.

To identify the possible cause and the circumstances of the error, refer to the Dr.Web for UNIX Internet Gateways log (by default, it is located in the `/var/log/syslog` file or the `/var/log/messages` file, depending on OS). Also, you can use the [command](#) `drweb-ctl log`.

Resolving the error

1. Check the path to the `drweb-meshd` executable file. Change the path, if necessary (the `ExePath` parameter in the [MeshD] [section](#) of the [configuration file](#)).

You can use the [commands](#) of the command-line management tool.

To view the current parameter value, execute the command:

```
$ drweb-ctl cfshow MeshD.ExePath
```

To set a new parameter value, execute the command:

```
# drweb-ctl cfset MeshD.ExePath <new path>
```

To restore the parameter value to the default, execute the command:

```
# drweb-ctl cfset MeshD.ExePath -r
```

2. If the configuration does not contain the settings for Dr.Web MeshD component or if the error persists after entering the correct path, install or reinstall the `drweb-meshd` package.

For details on how to install and uninstall Dr.Web for UNIX Internet Gateways or its components, refer to sections [Installing Dr.Web for UNIX Internet Gateways](#) and [Uninstalling Dr.Web for UNIX Internet Gateways](#).

If the error persists, contact [technical support](#) and be ready to name the error code.



Error message: *LookupD is not available*

Error code: x115

Internal designation: EC_NO_LOOKUPD

Description: Dr.Web LookupD component required for retrieving data from external sources is missing.

To identify the possible cause and the circumstances of the error, refer to the Dr.Web for UNIX Internet Gateways log (by default, it is located in the `/var/log/syslog` file or the `/var/log/messages` file, depending on OS). Also, you can use the [command](#) `drweb-ctl log`.

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 can use the [commands](#) of the command-line management tool.

To view the current parameter value, execute the 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 the 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 Dr.Web for UNIX Internet Gateways or its components, refer to sections [Installing Dr.Web for UNIX Internet Gateways](#) and [Uninstalling Dr.Web for UNIX Internet Gateways](#).

If the error persists, contact [technical support](#) and be ready to name the error code.

Error message: *UrlCheck is not available*

Error code: x116

Internal designation: EC_NO_URL_CHECK

Description: Dr.Web URL Checker component required for checking URL connections for belonging to potentially dangerous and unwanted categories is missing.

To identify the possible cause and the circumstances of the error, refer to the Dr.Web for UNIX Internet Gateways log (by default, it is located in the `/var/log/syslog` file or the `/var/log/messages` file, depending on OS). Also, you can use the [command](#) `drweb-ctl log`.



Resolving the error

1. Check the path to the `drweb-urlcheck` executable file. Change the path, if necessary (the `ExePath` parameter in the `[URLCheck]` section of the configuration file).

You can use the [commands](#) of the command-line management tool.

To view the current parameter value, execute the command:

```
$ drweb-ctl cfshow URLCheck.ExePath
```

To set a new parameter value, execute the command:

```
# drweb-ctl cfset URLCheck.ExePath <new path>
```

To restore the parameter value to the default, execute the command:

```
# drweb-ctl cfset URLCheck.ExePath -r
```

2. If the configuration does not contain the settings for Dr.Web URL Checker component or if the error persists after entering the correct path, install or reinstall the `drweb-urlcheck` package.

For details on how to install and uninstall Dr.Web for UNIX Internet Gateways or its components, refer to sections [Installing Dr.Web for UNIX Internet Gateways](#) and [Uninstalling Dr.Web for UNIX Internet Gateways](#).

If the error persists, contact [technical support](#) and be ready to name the error code.

Error message: *GateD is not available*

Error code: `x117`

Internal designation: `EC_NO_GATED`

Description: SplDer Gate component required for scanning network connections is missing.

To identify the possible cause and the circumstances of the error, refer to the Dr.Web for UNIX Internet Gateways log (by default, it is located in the `/var/log/syslog` file or the `/var/log/messages` file, depending on OS). Also, you can use the [command](#) `drweb-ctl log`.

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 can use the [commands](#) of the command-line management tool.

To view the current parameter value, execute the 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 Dr.Web for UNIX Internet Gateways or its components, refer to sections [Installing Dr.Web for UNIX Internet Gateways](#) and [Uninstalling Dr.Web for UNIX Internet Gateways](#).

If the error persists, contact [technical support](#) and be ready to name the error code.

Error message: *ScanEngine is not available*

Error code: x119

Internal designation: EC_NO_SCAN_ENGINE

Description: Dr.Web Scanning Engine component required for threat detection is missing or failed to start.

To identify the possible cause and the circumstances of the error, refer to the Dr.Web for UNIX Internet Gateways log (by default, it is located in the `/var/log/syslog` file or the `/var/log/messages` file, depending on OS). Also, you can use the [command](#) `drweb-ctl log`.

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 can use the [commands](#) of the command-line management tool.

To view the current parameter value, execute the 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 Dr.Web for UNIX Internet Gateways and receive a license. After receiving the license, check whether the [key file](#) is available and install it, if necessary.

- 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 Dr.Web for UNIX Internet Gateways or its components, refer to sections [Installing Dr.Web for UNIX Internet Gateways](#) and [Uninstalling Dr.Web for UNIX Internet Gateways](#).

If the error persists, contact [technical support](#) and be ready to name the error code.

Error message: *FileCheck is not available*

Error code: x120

Internal designation: EC_NO_FILE_CHECK

Description: Dr.Web File Checker component required for threat detection is missing or failed to start.

To identify the possible cause and the circumstances of the error, refer to the Dr.Web for UNIX Internet Gateways log (by default, it is located in the `/var/log/syslog` file or the `/var/log/messages` file, depending on OS). Also, you can use the [command](#) `drweb-ctl log`.

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 can use the [commands](#) of the command-line management tool.

To view the current parameter value, execute the 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 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 Dr.Web for UNIX Internet Gateways or its components, refer to sections [Installing Dr.Web for UNIX Internet Gateways](#) and [Uninstalling Dr.Web for UNIX Internet Gateways](#).

If the error persists, contact [technical support](#) and be ready to name the error code.



Error message: *ESAgent is not available*

Error code: x121

Internal designation: EC_NO_ESAGENT

Description: Dr.Web ES Agent component necessary to connect to the centralized protection server is missing.

To identify the possible cause and the circumstances of the error, refer to the Dr.Web for UNIX Internet Gateways log (by default, it is located in the `/var/log/syslog` file or the `/var/log/messages` file, depending on OS). Also, you can use the [command](#) `drweb-ctl log`.

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 can use the [commands](#) of the command-line management tool.

To view the current parameter value, execute the 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 Dr.Web for UNIX Internet Gateways or its components, refer to sections [Installing Dr.Web for UNIX Internet Gateways](#) and [Uninstalling Dr.Web for UNIX Internet Gateways](#).

If the error persists, contact [technical support](#) and be ready to name the error code.

Error message: *Firewall is not available*

Error code: x122

Internal designation: EC_NO_FIREWALL

Description: Dr.Web Firewall for Linux component required for scanning network connections is missing.

To identify the possible cause and the circumstances of the error, refer to the Dr.Web for UNIX Internet Gateways log (by default, it is located in the `/var/log/syslog` file or the `/var/log/messages` file, depending on OS). Also, you can use the [command](#) `drweb-ctl log`.



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 can use the [commands](#) of the command-line management tool.

To view the current parameter value, execute the 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 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 Dr.Web for UNIX Internet Gateways or its components, refer to sections [Installing Dr.Web for UNIX Internet Gateways](#) and [Uninstalling Dr.Web for UNIX Internet Gateways](#).

If the error persists, contact [technical support](#) and be ready to name the error code.

Error message: *NetChecker is not available*

Error code: x123

Internal designation: EC_NO_NET_CHECK

Description: Dr.Web Network Checker component required for scanning the downloaded files is missing.

To identify the possible cause and the circumstances of the error, refer to the Dr.Web for UNIX Internet Gateways log (by default, it is located in the `/var/log/syslog` file or the `/var/log/messages` file, depending on OS). Also, you can use the [command](#) `drweb-ctl log`.

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 can use the [commands](#) of the command-line management tool.

To view the current parameter value, execute the 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 Dr.Web for UNIX Internet Gateways or its components, refer to sections [Installing Dr.Web for UNIX Internet Gateways](#) and [Uninstalling Dr.Web for UNIX Internet Gateways](#).

If the error persists, contact [technical support](#) and be ready to name the error code.

Error message: *CloudD is not available*

Error code: x124

Internal designation: EC_NO_CLOUDD

Description: Dr.Web CloudD required for making requests to the Dr.Web Cloud service is missing.

To identify the possible cause and the circumstances of the error, refer to the Dr.Web for UNIX Internet Gateways log (by default, it is located in the `/var/log/syslog` file or the `/var/log/messages` file, depending on OS). Also, you can use the [command](#) `drweb-ctl log`.

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 can use the [commands](#) of the command-line management tool.

To view the current parameter value, execute the 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 Dr.Web for UNIX Internet Gateways or its components, refer to sections [Installing Dr.Web for UNIX Internet Gateways](#) and [Uninstalling Dr.Web for UNIX Internet Gateways](#).

If the error persists, contact [technical support](#) and be ready to name the error code.

Error message: *Unexpected error*



Error code: x125

Internal designation: EC_UNEXPECTED_ERROR

Description: An unexpected error has occurred in operation of one of the components.

To identify the possible cause and the circumstances of the error, refer to the Dr.Web for UNIX Internet Gateways log (by default, it is located in the `/var/log/syslog` file or the `/var/log/messages` file, depending on OS). Also, you can use the [command](#) `drweb-ctl log`.

Resolving the error

1. Restart Dr.Web for UNIX Internet Gateways by entering the command:

```
# service drweb-configd restart
```

If the error persists, contact [technical support](#) and be ready to name the error code.

Errors Without Code

Symptoms: Such components as Dr.Web ClamD, SpIDer Gate, Dr.Web ICAPD 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 exceeded the limit 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).



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 command:

```
# service drweb-configd restart
```

If the error persists, contact [technical support](#).



Symptoms: The web browser cannot establish connection to Dr.Web management web interface; the 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 an error message, or has no effect, install `drweb-configd` component (package) separately.



Also this may mean that PAM authentication is not used in the system. If so, please install and configure PAM (Dr.Web for UNIX Internet Gateways 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 Dr.Web for UNIX Internet Gateways or its components, refer to sections [Installing Dr.Web for UNIX Internet Gateways](#) and [Uninstalling Dr.Web for UNIX Internet Gateways](#).

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 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 command:

```
# iptables-save | grep -v "comment --comment --comment" | iptables-restore
```



The `iptables-save` and `iptables-restore` commands require the root privileges. To elevate your privileges, you can use the `su` and `sudo` commands.

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 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](#).



Appendix G. List of Abbreviations

The following abbreviations were used in this manual without further interpretation:

Convention	Complete form
<i>AD</i>	Microsoft Active Directory
<i>DN</i>	(LDAP) Distinguished Name
<i>FQDN</i>	Fully Qualified Domain Name
<i>GID</i>	Group ID (system user group identifier)
<i>GNU</i>	GNU project (GNU is Not Unix)
<i>HTML</i>	HyperText Markup Language
<i>HTTP</i>	HyperText Transfer Protocol
<i>HTTPS</i>	HyperText Transfer Protocol Secure (via SSL/TLS)
<i>ICAP</i>	Internet Content Adaptation Protocol
<i>ID</i>	Identifier
<i>IP</i>	Internet Protocol
<i>LDAP</i>	Lightweight Directory Access Protocol
<i>MBR</i>	Master Boot Record
<i>OID</i>	(SNMP) Object ID
<i>OS</i>	Operating System
<i>PID</i>	Process ID (system process identifier)
<i>PAM</i>	Pluggable Authentication Modules
<i>RPM</i>	Red Hat Package Manager
<i>RRA</i>	Round-Robin Archive
<i>RRD</i>	Round-Robin Database
<i>SNI</i>	Server Name Indication
<i>SNMP</i>	Simple Network Management Protocol
<i>SP</i>	Service Pack



Convention	Complete form
<i>SSH</i>	Secure Shell
<i>SSL</i>	Secure Sockets Layer
<i>TCP</i>	Transmission Control Protocol
<i>TLS</i>	Transport Layer Security
<i>UID</i>	User ID (system user identifier)
<i>URI</i>	Uniform Resource Identifier
<i>URL</i>	Uniform Resource Locator
<i>VBR</i>	Volume Boot Record



Index

A

- Abbreviations 460
- About the anti-virus 10
- Activation 61
- Appendices 372
- appendix
 - computer threat types 372
 - fighting computer threats 376

B

- Brief Instructions 75

C

- Centralized protection 22
- ClamD: configuration 222
- CloudD: configuration 352
- Command-line management 86
- Components 12
- computer threats 372
- ConfigD: configuration 83
- Configuration file 381
- Configuration parameters 381
- Configuring Security Subsystems 57
- Configuring SELinux 57
- Console uninstaller 45
- Console-based installer 35
- Conventions 9
- Custom Installation 51

D

- Database Update 43
- Dr.Web ClamD 221
- Dr.Web CloudD 351
- Dr.Web ConfigD 80
- Dr.Web Ctl 86
- Dr.Web ES Agent 261
- Dr.Web File Checker 231
- Dr.Web Firewall for Linux 180
- Dr.Web HTTPD 265
- Dr.Web ICAPD 137
- Dr.Web ICAPD Lua API 157
- Dr.Web License 30
- Dr.Web LookupD 354
- Dr.Web MeshD 340
- Dr.Web Network Checker 235

- Dr.Web Scanning Engine 246
- Dr.Web SNMP MIB 307
- Dr.Web SNMPD 293
- Dr.Web StatD 369
- Dr.Web Updater 252
- Dr.Web URL Checker 348
- drweb-clamd 221
- drweb-cloudd 351
- drweb-configd 80
- drweb-ctl 86
- drweb-ctl usage examples 117
- drweb-esagent 261
- drweb-filecheck 231
- drweb-firewall 180
- drweb-gated 175
- drweb-httpd 265
- drweb-icapd 137
- drweb-lookupd 354
- drweb-meshd 340
- drweb-netcheck 235
- drweb-netcheck: scanning cluster 242
- drweb-se 246
- drweb-snmpd 293
- drweb-statd 369
- drweb-update 252

E

- EICAR 64
- ESAgent: configuration 262

F

- fighting computer threats 376
- File access rights 21
- File permissions 21
- FileCheck: configuration 233
- Functions 10

G

- GateD: configuration 178
- General configuration settings 83
- Generating certificates 405
- Getting Started 61

H

- How to change settings 75
- How to Connect to the Centralized Protection Server 75



Index

How to install a key 75
How to protect a server 75
How to upgrade the product 75
How to view log 75
How to view settings 75
HTTP API 270
HTTP messages processing Lua procedure 157
HTTPD: configuration 267

I

ICAPD: configuration 141
Installation from .run package 33
Installation from distribution 33
Installation from native packages 35
Installation from universal packages 33
Installing from the Repository 35
Installing the anti-virus 31
Installing the Product 32
Integration with clamd clients intended for ClamAV 228
Integration with SNMP Monitoring Systems 299
Integration with Squid 66
Introduction 8
Isolation 20

K

Key File 61
Known errors 408

L

License key file 61
Licensing 30
LinuxFirewall: configuration 186
List of Supported Operating System Distributions 25
LookupD: configuration 356
Lua API Dr.Web Firewall for Linux 212

M

Management Web Interface 122
MeshD: configuration 344
Mobile mode 22
Modules 12
Monitoring systems 299
Monitoring via SNMP 299

N

NetCheck: configuration 237

O

Operating systems 25
Operation modes 22

P

Problems of SELinux 57
Product files 48
Product packages 48
Protection of a web server 70
Proxy mode of SplDer Gate 72

Q

Quarantine 20
Quarantine Directories 20

R

Registration 61
Removing the Product 44
Rules for Traffic Monitoring 385

S

ScanEngine: configuration 250
Section [ClamD] 222
Section [CloudD] 352
Section [ESAgent] 262
Section [FileCheck] 233
Section [GateD] 178
Section [HTTPD] 267
Section [ICAPD] 141
Section [LinuxFirewall] 186
Section [LookupD] 356
Section [MeshD] 344
Section [NetCheck] 237
Section [Root] 83
Section [ScanEngine] 250
Section [SNMPD] 295
Section [StatD] 370
Section [Update] 254
Section [Urlcheck] 350
SELinux Security 57
SNMPD: configuration 295
SplDer Gate 175
SSL CA 405
SSL certificates 405
SSL private keys 405



Index

Standalone mode 22
Starting command-line tool 88
Starting uninstaller 44
StatD: configuration 370
Structure of the product 12
System Requirements 25

T

Tasks 10
Technical support 378
Testing the Anti-virus 64
The procedures for traffic monitoring in Lua 212

U

Uninstalling distribution 44
Uninstalling from repository 45
Uninstalling native packages 45
Uninstalling the anti-virus 31
Uninstalling the Product 44
Update: configuration 254
Upgrading components 39
Upgrading the Product 38
Upgrading to a Newer Version 40
Urlcheck: configuration 350

W

Ways to install 32

