MAN PAGE

RAVUPDATE configuration file

Release Date: December 17, 2002

Product version: 8.4.1

Document revision: 2.0

Address: 223, Mihai Bravu Blvd, 3rd district, Bucharest, Romania
Phone / Fax: +40-21-321.78.03, Hotline: +40-21-321.78.59

MAN PAGE

Terms and conditions of the License Agreement

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

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

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

MAN PAGE

CONTENTS

RAVUPDATE configuration file

NAME

ravupdate - Update utility for **ravmd**.

SYNOPSIS

ravupdate [-adfhStTvVY] [--help] [--ftp-passive=on|off] [--usage] [--use-proxy=on|off] [--dump] [--serverresponse] [--tries=number] [--proxy=http://[user[:password]@]sever:port] [--timeout=secs] [--verbose] [--version]

DESCRIPTION

With new viruses appearing daily, your antivirus software (no matter what company is producing it) will become obsolete in weeks or even days, being unable to protect you from new threats. Updating your antivirus software is therefore a pre-requisite in the fight against malwares.

ravupdate is a utility helping you securely update your **RAV AntiVirus for Mail Servers** (**ravmd**). It has the following features:

- Works with the **ftp** and **http** (with and without proxy) protocols;
- Allows updating from mirrors;
- Guarantees the integrity of the downloaded files;
- Allows "on the fly" update;
- Auto-update;
- Default configuration.

Supported protocols

ravupdate currently works with the **ftp** and **http** (with and without proxy) protocols. If you have any suggestions concerning new protocols to be supported by **ravmd**, please send us an e-mail (ravteam@ravantivirus.com).

FTP

The FTP protocol allows the update procedure to be executed via one ftp server or one HTTP proxy.

The following options are customisable:

- User name;

- Password
- Server;
- Transfer type (active/passive);
- Timeout;
- Number of tries (in case of failure – how many times **ravupdate** will retry to execute the update procedure before returning an error message).

HTTP

The FTP protocol allows the update procedure to be executed via HTTP (with or without proxy).

The following options are customisable:

- User name;
- Password
- Server;
- Transfer type (active/passive)
- Timeout

Number of tries (in case of failure – how many times ravupdate will retry to execute the update procedure before returning an error message).

Updating from mirrors

ravupdate is designed to work with different servers implementing a certain type of protocol. This functionality was made possible using a minimal set of characteristics from each protocol.

For example, for a FTP protocol the mirror must implement the following commands: **USER, TYPE, PASV, PORT, RETR, REST, ABOR, QUIT, CWD.**

For HTTP, the web server (proxy) must support the 1.1 version of the HTTP protocol.

Integrity of the downloaded files

ravupdate guarantees the integrity of the updated files by downloading at first a file containing information about the package needing to be updated/downloaded. The file initially downloaded by **ravupdate** is called **infofile.**

The program uses the information contained in this infofile to find out what files will be updated. The initially downloaded file is also used to check the integrity of the other downloaded files. Only after all the files needed for the update procedure are correctly downloaded **ravupdate** moves to the next phase: installing the updated files.

MAN PAGE

“On the fly” update

If the product is working when the update procedure is undergone, **ravupdate** is trying its best to make sure that the impact of the update procedure on the programs using the updated files is minimal. For instance, in case the engine of RAV **AntiVirus for Mail Servers** is updated, **ravupdate** sends a SIGHUP signal to the **ravmd** daemon, forcing it to reload the updated engine. The programs already launched will continue the scanning process using the old engine, but new processes will make use of the updated engine. In case we are talking about updating the scanning daemon (quite rarely however), the daemon must be stopped and restarted using the updated program.

Auto-update

In case a bug is reported in **ravupdate** or a new version of this program is launched, the file containing info about the package to be updated also contains information about the version of **ravupdate**. The program will update itself, reload and start the update procedure for the selected package.

Default configuration

ravupdate is shipped with a default configuration that allows running the update procedure on most of the computers using RAV **AntiVirus for Mail Servers**. However, for an enhanced performance of **ravupdate**, we advise you to customize the configuration of **ravupdate** according to your specifications.

USAGE

Definitions

Module

Data containing information about one or more file logically grouped to obtain a certain functionality in RAV **AntiVirus for Mail Servers**.

Package

Data (transmitted in one communication session) containing one or more modules.

Example:

RAV for Qmail is a package that might contain the following modules:

- rav engine – containing the files from the rave folder
- ravmd – containing the ravmd file and the scripts allowing you to launch/stop/restart/reload **ravmd** while it is working.
- ravqmail – containing the RAV filter for Qmail, linking **ravmd** to Qmail.

Description

At the beginning, **ravupdate** downloads a file containing information about the package you want to be updated. Depending on the modules you selected to be updated and the information from this file, **RAV AntiVirus for Mail Serves** checks every file susceptible to be updated. A list containing the files needing to be updated is created and the downloading process for these files begins.

After checking the integrity of the downloaded files **ravupdate** begins the install process for each of the modules needing update, in a precise order. In the example given above, first the updated engine is installed and then the updated **ravmd** and the filer are installed.

The installation procedure contains three different stages. For some modules, depending on their peculiarities, one or two of these stages might be skipped. However, the order these stages are executed is always as follows:

- **Pre-install**. The purpose of this step is to insure the successful of the install procedure. During this pre-install phase usually the correct permissions are checked and it is established if some programs have to be shut down.
- **Install**. The downloaded files are copied to a temporary location allowing the product to operate correctly.
- **Post-install**. **RAV AntiVirus for Mail Servers** is restarted or, in case the restart is not necessary, the user is notified the update procedure has successfully ended.

Example:

Here is a brief example of what each phase is doing in case of updating the **engine** module containing the scanning engine.

- During the *Pre-install* phase: **ravupdate** is checking if you have writing permissions for copying the downloaded files in the folder containing the old engine
- During the *Install* phase: **ravupdate** is copying the new files to the rave folder;
- During the *Post-install* phase: **ravupdate** is sending a SIGHUP signal to the ravmd process (if active), forcing it to reload the new engine.

Security and ravupdate

As far as the process security is concerned, we have to keep in mind that, in most cases, the update is made for a product under execution. Therefore, it is very important to ensure the authentication and the integrity of the files being updated. **ravupdate** answers this demanding request using the following mechanism:

- The **infofile** is digitally signed using an asymmetric system. The public key is encoded in the update program. This update program is only using files signed with the corresponding private key.
- The **infofile** contains the following information about the files used in the update process:
  - The file’s size;
  - The date of the last modification in the file;

- The path of the file (on the server used for updating **RAV AntiVirus for Mail Servers**);
- The path on the file (on the local disk, where the update is being executed);
- A digital signature of the file.

The update process uses this info to check each file downloaded from the update mirror in terms of precision (*is this the file I have to download?*), integrity (*does the downloaded file have the size specified in the initially downloaded file*?) and identity (is the digital signature recognized?). Only if the checking is returning the expected values the update process itself is launched. If the checking procedure for the downloaded files fails, the updating process returns an error code. Usually, this means the file was not correctly downloaded from the mirror site or this site is just executing its resync procedure.

CONFIGURATION

ravupdate can be configured:

- using a configuration file;
- using command-line parameters.

Note: The command-line parameters are overwriting the parameters from the configuration file.

SECTION DESCRIPTION

The configuration file for **ravupdate** contains *sections* and these sections contain *attributes*, *names* and *values*.

The values can be: numbers, strings or Booleans.

The [global] section

The [global] section contains the attributes managing the general behaviour of **ravupdate**:

CONFDIR

Description: String containing the install path for the configuration files.

Default value: ${ETCDIR}.

DATADIR

Description: String containing the path to data required by **RAV AntiVirus** programs.

Default value: ${DATADIR}.

INSTALL

Description: String containing the install path for **RAV AntiVirus** programs.

MAN PAGE

Default value: /opt/rav for Linux, Solaris and MacOS X, /usr/local for BSDs.

Note1: Please note that the update procedure will return an error message in case this folder does not exist or the user does not have writing privileges in this folder.

Note 2: The INSTALL parameter has no correspondent in the command-line.

PLATFORM

Description: String describing the platform on which you wish to execute the update procedure. The available options are: i386, ppc, sparc, s390, etc.

The default value is usually correct.

Note 1: Selecting a different platform than the one on which the update procedure is executing might determine the update process to fail.

Note 2: The PLATFORM parameter has no correspondent in the command-line.

alarm

Description: Number specifying the time (in seconds) allowed for one process to end the update procedure.

Note 1: In case the program is not ending the update procedure in the specified period of time, the program will return a corresponding error message.

Note 2: The **alarm** parameter has no correspondent in the command-line.

proxy

Description: String with the following format: http://proxy_server_name:port_number (the port_number parameter may be overlooked) specifying the proxy server used in the update procedure.

Note 1: In case you wish to use a proxy, the use-proxy parameter has to be configured to Yes.

Note 2: The **proxy** parameter from the *config* file can be overwritten from the command-line using the following parameters (in short or long form): -p, --proxy=http://port:name

Note 3: When the port_number parameter is overlooked, ravupdate is using the default value (80).

proxy-auth

Description: String helping to authenticate the proxy user.

Default value: None. This is currently the only available value. Some other values will be implemented in future versions of **ravupdate**.

Note: The proxy-auth parameter can be overwritten from the command-line using: -a, --proxy-auth=none|basic|digest

serverresponse

Description: Boolean. When set to Yes the program will display the commands it is sending and the answers received from the server.

Note 1: The serverresponse parameter is useful to establish the potential causes for an eventual failure of the update procedure, so it is advisable to set it to Yes.

Note 2: The serverresponse parameter can be activated from the command line using: -S, --server-response

temppath

Description: String containing a valid path for the folder used to save the temporary files downloaded from the mirror site.

Note 1: A file named update.start is created in the specified **temppath**. This file is used for synchronizing/blocking/using more than one update process. It is advisable that only update program has accessing right for this file.

Note 2: The temppath parameter has no correspondent in the command-line.

timeout

Description: Number specifying the time (in seconds) allowed for a writing/reading operation to end successfully.

Default value: 0 (no timeout).

Note 1: We do not advise you to set for this parameter values over 20.

Note 2: The **timeout** parameter can be overwritten from the command-line using: -T, --timeout=nr

tries

Description: Number specifying the number of tries allowed for the communication protocol before the update procedure fails.

Note 1: In case the value for tries is set to 0, the **alarm** parameter is the one responsible for establishing the time allowed for one process to complete.

Note 2: The **tries** parameter can be overwritten from the command-line using: -t, --tries=nr

use-proxy

Description: Boolean enabling/disabling the usage of the proxy server for one specific protocol.

Note: The **use-proxy** parameter can be enabled from the command-line using: -Y, --use-proxy

verbose

Description: Boolean allowing or not **ravupdate** to display more information about the process.

Note 1: The **verbose** parameter can be useful when debugging or studying the behaviour of **ravupdate**.

Note 2: The **verbose** parameter can be enabled from the command-line using: -v

The [ftp] section

The [ftp] section contains the following parameters used for the configuring the **ftp** connections: **passive**, **password**, **port**, **username**.

passive

Description: Boolean establishing if the update program or the **ftp** server is initiating the connection.

Available values: Off/On.

Default value: Off.

The **ftp** protocol uses two connections for transferring files between the **ftp server** and the **ftp client**. The first connection is used to send commands and receive answers. The second connection is created for each transferred file. Closing this second connection usually marks the ending the file transfer. For this second connection, the **ftp server** or the **ftp client** has to initiate the connection and send information, in order to allow the other to connect.

The **passive** parameter is managing this connection:

- When set to On, the update program is establishing the connection with the ftp server and is sending the information the ftp server needs for connecting.
- When set to Off, the ftp server is establishing the connection and send the client the available information in the first transmission.

Note 1: Some ftp servers are using passive connections; other ftp servers are using active connections. Please make sure you have all the required information before configuring the **passive** parameter.

Note 2: The passive parameter can be overwritten from the command-line using: -f, --ftp-passive=on|off

password

Description: String used by the **ftp** server to identify the user executing the update process.

The *default value* for **password** depends on the version of the program you are using.

Note: The password parameter has no correspondent in the command-line.

port

Description: Number used to specify the default port used by the **ftp** protocol.

Default value: 21.

username

Description: String used by the **ftp** server to identify the user executing the update process.

Default value: **ftp**.

Note: The **username** parameter has no correspondent in the command-line.

MAN PAGE

The [http] section

The [http] section contains the following parameters used for the configuring the http connections: password, port, username.

password

Description: String used by the http server to identify the user executing the update process.

Default value: ravupdate(snapshot-20020528)@ravantivirus.com.

Note: The password parameter has no correspondent in the command-line.

port

Description: Number used to specify the default port used by the http protocol.

Default value: 80.

username

Description: String used by the http server to identify the user executing the update process.

Default value: anonymous.

Note: The username parameter has no correspondent in the command-line.

The [host] section

The [host] section contains information about the infofile.

host

Description: String representing the name or the IP address of the server used for updating RAV AntiVirus for Mail Servers.

Note: The host parameter has to be customized to the closest server.

infofile

Description: String containing the relative path to infofile.

protocol

Description: String containing the name of the protocol used for the updating process.

Default value: ftp.

Note: The protocol parameter has no correspondent in the command-line.

The [modules] section

engine

Description: Boolean specifying if the user wants to update only the engine of RAV AntiVirus for Mail Servers.

Available options: **On/Off.**

Default value: On.

Important: The [modules] section will contain other parameters ('ravcgate', 'ravdmail', 'ravexim', 'ravmd', 'RAVMilter', 'ravpostfix', 'ravqmail' and 'ravsendmail') for specifying if the modules for rav clients will also be updated. These options in are not yet implemented. The *default value* for these parameters is **Off**.

EXIT STATUS

0	Update has been successfully completed.
1	There are no files to update - you have latest files.
2	Configuration error - check your **rup.conf** file
3	Another **ravupdate** process is running.
4	Can't find the temporary directory or the temporary directory does not exist.
5	System error (e.g fork, malloc etc).
6	Other error. Please manually run **ravupdate** with -S and -v to see what is wrong.

FILES

${ETCDIR}/rup.conf

BUGS

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

SEE ALSO

ravmd(8)