SAVAPI
API REFERENCE MANUAL

Table of Contents

1 SAVAPI

SAVAPI stands for Secure AntiVirus Application Programming Interface. Its main purpose is to offer a very simple scanning interface for clients who want to programmatically integrate scanning services into their applications.

This document explains how to use the SAVAPI interface shared libraries written and provided by Avira. The provided LICENSE file explains the terms and conditions for using the SAVAPI interface shared libraries.

See: SAVAPI options , SAVAPI constants , SAVAPI defines , SAVAPI structures , SAVAPI functions

2 Module Index

2.1 Modules

Here is a list of all modules:

3 Data Structure Index

3.1 Data Structures

Here are the data structures with brief descriptions:

4 File Index

4.1 File List

Here is a list of all files with brief descriptions:

5 Module Documentation

5.1 SAVAPI constants

5.1.1 Modules

- SAVAPI return codes
- Error categories
- Error levels
- Initialization flags
- Scan warnings
- Iframes informations
- Scan informations
- SAVAPI options
- Callbacks' ids
- SAVAPI report scan details types

- SAVAPI report content types
- SAVAPI signals
- SAVAPI commands
- SAVAPI scan statuses
- File types

5.2 SAVAPI return codes

5.2.1 Defines

- #define SAVAPI3_S_OK 0
  Operation ended with success.
- #define SAVAPI3_E_INVALID_PARAMETER 1
  One of supplied parameters is invalid.
- #define SAVAPI3_E_ALREADY_INITIALIZED 2
  SAVAPI was already initialized.
- #define SAVAPI3_E_NOT_INITIALIZED 3
  SAVAPI is not initialized.
- #define SAVAPI3_E_BUFFER_TOO_SMALL 4
  Supplied buffer is too small.
- #define SAVAPI3_E_CONNECTION_MODE_NOT_SET 5
  Connection mode flag is not set.
- #define SAVAPI3_E_HOSTNAME_NOT_SET 6
  Host name is not set.
- #define SAVAPI3_E_NO_MEMORY 7
  Memory allocation failed.
- #define SAVAPI3_E_VDF_NOT_FOUND 8
  VDF file(s) not found.
- #define SAVAPI3_E_VDF_READ 9
  VDF file(s) read failed.
- #define SAVAPI3_E_VDF_CRC 10
  VDF file(s) crc check failed.
- #define SAVAPI3_E_VDF_VERSION 11
  Inconsistent versions in VDF files set.
- #define SAVAPI3_E_WRONG_ENGINE 12
  Engine initialization failed.
- #define SAVAPI3_E_ENGINE_NOT_FOUND 13
  Engine file(s) not found.
- #define SAVAPI3_E_WRONG_SAVAPI 14
  Invalid SAVAPI binary encountered.
- #define SAVAPI3_E_SELFCHK_PATCHED 15
  Inconsistent versions in engine files set.
- #define SAVAPI3_E_SELFCHK_FILE_ERR 16
  Engine file(s) read failed.
- #define SAVAPI3_E_SELFCHK_FILE_CRC 17
  Engine file(s) crc check failed.
- #define SAVAPI3_E_KEYFILE 18

Keyfile error.

- #define SAVAPI3_E_INTERNAL 19
  SAVAPI internal error.
- #define SAVAPI3_E_NOT_SUPPORTED 20
  Unsupported feature.
- #define SAVAPI3_E_RESULT_ERROR 21
  An error occurred during a file scan.
- #define SAVAPI3_E_RESULT_FILE_NOT_FOUND 22
  Could not extract file.
- #define SAVAPI3_E_OPTION_NOT_SUPPORTED 23
  Option is not supported.
- #define SAVAPI3_E_HIT_MAX_REC 24
  Archive maximum recursion limit reached.
- #define SAVAPI3_E_HIT_MAX_SIZE 25
  Archive maximum extraction size reached.
- #define SAVAPI3_E_HIT_MAX_RATIO 26
  Archive maximum extraction ratio reached.
- #define SAVAPI3_E_ENCRYPTED 27
  Encrypted contents found.
- #define SAVAPI3_E_UNSUPPORTED 28
  Unsupported archive type/format.
- #define SAVAPI3_E_PROC_ERROR 29
  Archive generic processing error.
- #define SAVAPI3_E_INCOMPLETE 30
  File was not completely scanned.
- #define SAVAPI3_E_PARTIAL 31
  Cannot extract multi-volume archive.
- #define SAVAPI3_E_HIT_MAX_COUNT 32
  Maximum number of files in archive reached.
- #define SAVAPI3_E_ABORTED 33
  Scan was aborted by signal.
- #define SAVAPI3_E_TIMEOUT 34
  Scan timed out.
- #define SAVAPI3_E_RESULT_SUSPICIOUS 35
  Possible infected file.
- #define SAVAPI3_E_DECRYPT 36
  Could not decrypt virus.
- #define SAVAPI3_E_SECTOR_READ 37
  Read error (boot record access).
- #define SAVAPI3_E_SECTOR_WRITE 38
  Write error (boot record access).
- #define SAVAPI3_E_SECTOR_INVALID 39
  Invalid sector (no bios signature) (boot record access).
- #define SAVAPI3_E_FILE_OPEN 40
  Could not open file.
- #define SAVAPI3_E_FILE_READ 41
  Could not read file.

- #define SAVAPI3_E_FILE_WRITE 42
  Could not write file.
- #define SAVAPI3_E_DEMOMODE 43
  SAVAPI in DEMO mode. Call not executed.
- #define SAVAPI3_E_QUERY_DISK_PARAM 44
  Problem while getting disk geometry.
- #define SAVAPI3_E_FILE_LEN 45
  Wrong file size in directory.
- #define SAVAPI3_E_FILE_DATE 46
  Invalid file date.
- #define SAVAPI3_E_FILE_DAMAGED 47
  Possible corrupted file.
- #define SAVAPI3_E_RESULT_DROPPER 48
  Macro heuristic: possible dropper.
- #define SAVAPI3_E_RESULT_TROJAN 49
  Macro heuristic: possible trojan horse.
- #define SAVAPI3_E_RESULT_POLYMORPHIC 50
  Macro heuristic: possible polymorphic virus.
- #define SAVAPI3_E_FORCE_BACKUP 51
  MBS is ok, force a backup to user.
- #define SAVAPI3_E_PARTITION_TABLE 52
  Partition tables unequal.
- #define SAVAPI3_E_RESULT_BOOTIMAGE 53
  File contains a boot virus image.
- #define SAVAPI3_E_FILE_PACKED 54
  File is packed PKLite or LZExe.
- #define SAVAPI3_E_FILE_OLE 55
  File is a compound doc (OLE2).
- #define SAVAPI3_E_FILE_TEMPLATE 56
  File contains a word template.
- #define SAVAPI3_E_FILE_MACRO 57
  File contains macros.
- #define SAVAPI3_E_FILE_ARCHIVE 58
  File is an archive.
- #define SAVAPI3_E_SECTOR_KNOWN 59
  Known good boot sector.
- #define SAVAPI3_E_SECTOR_UNKNOWN 60
  Unknown boot sector.
- #define SAVAPI3_E_SECTOR_CONSTANT 61
  Boot sector contains constant data.
- #define SAVAPI3_E_NOT_UPTODATE 62
  SAVAPI is not up to date.
- #define SAVAPI3_E_SETUP_PRODUCT 63
  SAVAPI product is not set.
- #define SAVAPI3_E_NO_PARAMETER 64
  No parameter given to command.
- #define SAVAPI3_E_INVALID_VALUE 65

Invalid value in configuration or command.

- #define SAVAPI3_E_CHDIR_FAILED 66
  Could not change directory.
- #define SAVAPI3_E_NOT_ABSOLUTE_PATH 67
  Path is not absolute.
- #define SAVAPI3_E_DIR_NOT_EXISTS 68
  Directory path does not exist.
- #define SAVAPI3_E_MATCHED 69
  File was filtered from scanning.
- #define SAVAPI3_E_CONVERSION_FAILED 70
  Converting failed.
- #define SAVAPI3_E_FILE_OFFICE 71
  Office document found.
- #define SAVAPI3_E_FILE_IN_ARCHIVE 72
  Filename from archive.
- #define SAVAPI3_E_CONNECTION_FAILED 73
  Connection with the SAVAPI Service failed.
- #define SAVAPI3_E_RECEIVE_FAILED 74
  Failed to receive data from the SAVAPI Service.
- #define SAVAPI3_E_SEND_FAILED 75
  Failed to send data to the SAVAPI Service.
- #define SAVAPI3_E_OPTION_VALUE_INVALID 76
  Invalid option value.
- #define SAVAPI3_E_REPAIR_FAILED 77
  Repair an infected file failed.
- #define SAVAPI3_E_FILE_CREATE 78
  Failed to create file.
- #define SAVAPI3_E_FILE_DELETE 79
  Failed to delete file.
- #define SAVAPI3_E_FILE_CLOSE 80
  Failed to close file.
- #define SAVAPI3_E_UNKNOWN 81
  Unknown engine error.
- #define SAVAPI3_E_PREFIX_SET 90
  Failed to set a detect type option.
- #define SAVAPI3_E_PREFIX_GET 91
  Failed to retrieve a detect type option.
- #define SAVAPI3_E_INVALID_QUERY 92
  Invalid query for SAVAPI Service.
- #define SAVAPI3_E_KEY_NO_KEYFILE 101
  Keyfile has not been found.
- #define SAVAPI3_E_KEY_ACCESS_DENIED 102
  Access to key file has been denied.
- #define SAVAPI3_E_KEY_INVALID_HEADER 103
  An invalid header has been found.
- #define SAVAPI3_E_KEY_KEYFILE_VERSION 104
  Invalid keyfile version number.

- #define SAVAPI3_E_KEY_NO_LICENSE 105
  No valid license found.
- #define SAVAPI3_E_KEY_FILE_INVALID 106
  Key file is invalid (invalid CRC).
- #define SAVAPI3_E_KEY_RECORD_INVALID 107
  Invalid license record detected.
- #define SAVAPI3_E_KEY_EVAL_VERSION 108
  Application is evaluation version.
- #define SAVAPI3_E_KEY_DEMO_VERSION 109
  Application is demo version.
- #define SAVAPI3_E_KEY_ILLEGAL_LICENSE 110
  Illegal (cracked) license in keyfile.
- #define SAVAPI3_E_KEY_NO_FUP_LICENSE 111
  No FUP II/III license found.
- #define SAVAPI3_E_KEY_NO_FUP2_KEYFILE 112
  No FUP II/III keyfile found.
- #define SAVAPI3_E_KEY_EXPIRED 113
  This key has expired.
- #define SAVAPI3_E_KEY_READ 114
  Error reading from key file.
- #define SAVAPI3_E_LICENSE_RESTRICTION 120
  Operation not allowed (license restriction).
- #define SAVAPI3_E_LOADING_ENGINE_MODULES 121
  Error loading engine modules.
- #define SAVAPI3_E_BUSY 122
  SAVAPI is busy.
- #define SAVAPI3_E_ENCRYPTED_MIME 123
  Encrypted mail found.
- #define SAVAPI3_E_NON_ADDRESSABLE 124
  Non addressable memory location.
- #define SAVAPI3_E_MEMORY_LIMIT 125
  Internal memory limit reached.
- #define SAVAPI3_E_PROC_INCOMPLETE_BLOCK_READ 150
  Incomplete archive block read.
- #define SAVAPI3_E_PROC_BAD_HEADER 151
  Bad archive header.
- #define SAVAPI3_E_PROC_INVALID_COMPRESSED_DATA 152
  Bad compressed data.
- #define SAVAPI3_E_PROC_OBSOLETE 153
  Obsolete archive information.
- #define SAVAPI3_E_PROC_BAD_FORMAT 154
  Bad header format.
- #define SAVAPI3_E_PROC_HEADER_CRC 155
  Bad header crc.
- #define SAVAPI3_E_PROC_DATA_CRC 156
  Bad data crc.
- #define SAVAPI3_E_PROC_FILE_CRC 157

Bad crc for extracted file.

- #define SAVAPI3_E_PROC_BAD_TABLE 158
  Invalid decompression table.
- #define SAVAPI3_E_PROC_UNEXPECTED_EOF 159
  Unexpected end of file.
- #define SAVAPI3_E_PROC_ARCHIVE_HANDLE 160
  Archive internal handle error.
- #define SAVAPI3_E_PROC_NO_FILES_TO_EXTRACT 161
  No files could be extracted.
- #define SAVAPI3_E_PROC_CALLBACK 162
  Archive internal callback error.
- #define SAVAPI3_E_PROC_TOTAL_LOSS 163
  File extraction failed.

5.2.2 Define Documentation

#define SAVAPI3_E_ABORTED 33

Scan was aborted by signal.

Note:
A scan in progress was aborted by user with SAVAPI3_SIGNAL_SCAN_ABORT signal.

#define SAVAPI3_E_ALREADY_INITIALIZED 2

SAVAPI was already initialized.

Note:
Trying to initialize an already initialized SAVAPI library (SAVAPI3_initialize was already called successfully).

#define SAVAPI3_E_BUFFER_TOO_SMALL 4

Supplied buffer is too small.

Note:
An interface function that requires a buffer size as parameter was called with a value smaller than the needed size.

#define SAVAPI3_E_BUSY 122

SAVAPI is busy.

Note:

A configuration request was given during scanning a file (for instance SET/GET command or callback register/unregister command). SAVAPI3_uninitialize was called without releasing all SAVAPI instances before.

#define SAVAPI3_E_CHDIR_FAILED 66

Could not change directory.

Note:

Failure in SAVAPI Client Library communication with SAVAPI Service resulting in an unsuccessful SET CWD command.
This error can only be triggered by the SAVAPI Client Library.

#define SAVAPI3_E_CONNECTION_FAILED 73

Connection with the SAVAPI Service failed.

Note:

The SAVAPI service is not running on the specified interface.
This error can only be triggered by the SAVAPI Client Library.

#define SAVAPI3_E_CONNECTION_MODE_NOT_SET 5

Connection mode flag is not set.

Note:

The SAVAPI3_INSTANCE_INIT::flags in the instance creation structure is not set to a known connection mode.
This error can only be triggered by the SAVAPI Client Library.

#define SAVAPI3_E_CONVERSION_FAILED 70

Converting failed.

Note:

A string could not be converted from one encoding to another (for instance a string could not be converted from SAVAPI_TCHAR to char, or in case of SAVAPI Client Library a string could not be converted from SAVAPI_TCHAR to the SAVAPI Service's text mode encoding).

#define SAVAPI3_E_DECRYPT 36

Could not decrypt virus.

Note:

Not used.

#define SAVAPI3_E_DEMOMODE   43

SAVAPI in DEMO mode. Call not executed.

Note:

Not used.

#define SAVAPI3_E_DIR_NOT_EXISTS   68

Directory path does not exist.

Note:

Path to a given directory does not exist (for example the path of the temporary scanning directory does not exist).
This error can only be triggered by the SAVAPI Client Library.

#define SAVAPI3_E_ENCRYPTED   27

Encrypted contents found.

Note:

One or more files inside the archive are encrypted, but there are also files which are not encrypted and can be extracted; or all files inside the archive are encrypted and it's not possible to extract them.

#define SAVAPI3_E_ENCRYPTED_MIME   123

Encrypted mail found.

Note:

While scanning an archive an encrypted mail was found.

#define SAVAPI3_E_ENGINE_NOT_FOUND   13

Engine file(s) not found.

Note:

One or more engine files are not present in the engine's directory.

#define SAVAPI3_E_FILE_ARCHIVE   58

File is an archive.

Note:

Not used.

#define SAVAPI3_E_FILE_CLOSE 80

Failed to close file.

Note:

Failed to close a temporary file in the temporary scanning directory because there are no access rights, file was accidentally deleted, etc.

#define SAVAPI3_E_FILE_CREATE 78

Failed to create file.

Note:

Failed to create a temporary file in the temporary scanning directory because there are no access rights, or the file already exists, etc.

#define SAVAPI3_E_FILE_DAMAGED 47

Possible corrupted file.

Note:

Not used.

#define SAVAPI3_E_FILE_DATE 46

Invalid file date.

Note:

Not used.

#define SAVAPI3_E_FILE_DELETE 79

Failed to delete file.

Note:

Failed to delete a temporary file in the temporary scanning directory because there are no access rights, file is locked, file does not exist anymore, etc.

#define SAVAPI3_E_FILE_IN_ARCHIVE 72

Filename from archive.

Note:

Not used.

#define SAVAPI3_E_FILE_LEN 45

Wrong file size in directory.

Note:
Not used.

#define SAVAPI3_E_FILE_MACRO 57

File contains macros.

Note:
Not used.

#define SAVAPI3_E_FILE_OFFICE 71

Office document found.

Note:
Not used.

#define SAVAPI3_E_FILE_OLE 55

File is a compound doc (OLE2).

Note:
Not used.

#define SAVAPI3_E_FILE_OPEN 40

Could not open file.

Note:
File is missing or there are no access rights to open it.

#define SAVAPI3_E_FILE_PACKED 54

File is packed PKLite or LZExe.

Note:
Not used.

#define SAVAPI3_E_FILE_READ 41

Could not read file.

File read error

Note:

There are no access rights to read file, or the file has been removed, or data from file end is missing, or file is truncated.

#define SAVAPI3_E_FILE_TEMPLATE 56

File contains a word template.

Note:

Not used.

#define SAVAPI3_E_FILE_WRITE 42

Could not write file.

Note:

There are no access rights to write file, or the file has been removed. Disk quota exceeded or disk is damaged.

#define SAVAPI3_E_FORCE_BACKUP 51

MBS is ok, force a backup to user.

Note:

Not used.

#define SAVAPI3_E_HIT_MAX_COUNT 32

Maximum number of files in archive reached.

Note:

Maximum files count limit was reached while scanning an archive. The scanning will be aborted as soon as the limit is exceeded.

#define SAVAPI3_E_HIT_MAX_RATIO 26

Archive maximum extraction ratio reached.

Note:

Size of an uncompressed file has exceeded the maximum extraction ratio. The decompression will be aborted as soon as the limit is exceeded.

#define SAVAPI3_E_HIT_MAX_REC 24

Archive maximum recursion limit reached.

Note:
The limit on the maximum number of archive recursions was exceeded when extracting a file because the file was packed too many times or it contained other deeply nested files. The decompression will be aborted as soon as the limit is exceeded.

#define SAVAPI3_E_HIT_MAX_SIZE 25

Archive maximum extraction size reached.

Note:
Size of an uncompressed file has exceeded the maximum extraction size. The decompression will be aborted as soon as the limit is exceeded.

#define SAVAPI3_E_HOSTNAME_NOT_SET 6

Host name is not set.

Note:
The SAVAPI3_INSTANCE_INIT::host_name field in the instance creation structure was not set. This error can only be triggered by the SAVAPI Client Library.

#define SAVAPI3_E_INCOMPLETE 30

File was not completely scanned.

Note:
Scanning was aborted by user or as result of a terminal warning or error.

#define SAVAPI3_E_INTERNAL 19

SAVAPI internal error.

Note:
An unexpected internal event prevented the normal execution of the library (incorrect pointers, incorrect return values, etc.). Normally this error should never occur. If this error occurs there is a major problem which must be fixed.

#define SAVAPI3_E_INVALID_PARAMETER 1

One of supplied parameters is invalid.

Note:

At least one of the function's parameters is invalid (invalid pointers, empty strings, out of range values, etc.).

#define SAVAPI3_E_INVALID_QUERY 92

Invalid query for SAVAPI Service.

Note:

Failure in SAVAPI Client Library communication with SAVAPI Service resulting in an unacceptable command (invalid command, syntax error).
This error can only be triggered by the SAVAPI Client Library

#define SAVAPI3_E_INVALID_VALUE 65

Invalid value in configuration or command.

Note:

Failure in SAVAPI Client Library communication with SAVAPI Service resulting in commands with invalid values which cannot be accepted by Service (for instance SET or GET commands with invalid values). The engine path given to the SAVAPI3_reload_engine_ex() function collides with previous engine path.

#define SAVAPI3_E_KEY_ACCESS_DENIED 102

Access to key file has been denied.

Note:

Not used.

#define SAVAPI3_E_KEY_DEMO_VERSION 109

Application is demo version.

Note:

Not used.

#define SAVAPI3_E_KEY_EVAL_VERSION 108

Application is evaluation version.

Note:

Not used.

#define SAVAPI3_E_KEY_EXPIRED 113

This key has expired.

Note:
Not used.

#define SAVAPI3_E_KEY_FILE_INVALID 106

Key file is invalid (invalid CRC).

Note:
Not used.

#define SAVAPI3_E_KEY_ILLEGAL_LICENSE 110

Illegal (cracked) license in keyfile.

Note:
Not used.

#define SAVAPI3_E_KEY_INVALID_HEADER 103

An invalid header has been found.

Note:
Not used.

#define SAVAPI3_E_KEY_KEYFILE_VERSION 104

Invalid keyfile version number.

Note:
Not used.

#define SAVAPI3_E_KEY_NO_FUP2_KEYFILE 112

No FUP II/III keyfile found.

Note:
Not used.

#define SAVAPI3_E_KEY_NO_FUP_LICENSE 111

No FUP II/III license found.

Note:

Not used.

#define SAVAPI3_E_KEY_NO_KEYFILE   101

Keyfile has not been found.

Note:

Not used.

#define SAVAPI3_E_KEY_NO_LICENSE   105

No valid license found.

Note:

Not used.

#define SAVAPI3_E_KEY_READ   114

Error reading from key file.

Note:

Not used.

#define SAVAPI3_E_KEY_RECORD_INVALID   107

Invalid license record detected.

Note:

Not used.

#define SAVAPI3_E_KEYFILE   18

Keyfile error.

Note:

Not used.

#define SAVAPI3_E_LICENSE_RESTRICTION   120

Operation not allowed (license restriction).

Note:

Scan command was issued without setting a valid product id.

#define SAVAPI3_E_LOADING_ENGINE_MODULES 121

Error loading engine modules.

Note:
SAVAPI could not load engine modules because they are not available or there are no access rights.

#define SAVAPI3_E_MATCHED 69

File was filtered from scanning.

Note:
File matched a black list rule and was not scanned.

#define SAVAPI3_E_MEMORY_LIMIT 125

Internal memory limit reached.

Note:
An engine-internal safety limit regarding memory usage of a subroutine has been reached (this can i.e. be caused by excessively large dictionaries in archives).

#define SAVAPI3_E_NO_MEMORY 7

Memory allocation failed.

Note:
There is not enough memory available for allocation.

#define SAVAPI3_E_NO_PARAMETER 64

No parameter given to command.

Note:
Failure in SAVAPI Client Library communication with SAVAPI service resulting in commands with no parameters (for instance SET or GET commands).
This error can only be triggered by the SAVAPI Client Library.

#define SAVAPI3_E_NON_ADDRESSABLE 124

Non addressable memory location.

Note:
A scan request was issued for an address that is not in the available address space for the current platform.
For example, on a 64 bit machine the available address space is [0..MAX_INT_64].

#define SAVAPI3_E_NOT_ABSOLUTE_PATH 67

Path is not absolute.

Note:
Path to a given or required directory is not absolute (for example the path of the temporary scanning directory is not absolute).

#define SAVAPI3_E_NOT_INITIALIZED 3

SAVAPI is not initialized.

Note:
The used functionality requires the SAVAPI library to be initialized first (a successful call of SAVAPI3_initialize is needed before).

#define SAVAPI3_E_NOT_SUPPORTED 20

Unsupported feature.

Note:
The requested functionality (feature, command, option) may be known but it is not supported by this version of SAVAPI or engine. For instance a called function is not available in the current library mode (SAVAPI3_is_running_ex() is not available in SAVAPI Library, or SAVAPI3_reload_engine_ex() is not available in SAVAPI Client Library); or the used signal id is unknown (the only known signal for SAVAPI3_send_signal is SAVAPI3_SIGNAL_SCAN_ABORT); or a new functionality was added but is not yet implemented or not supported yet by the current library version or within the current engine version.

#define SAVAPI3_E_NOT_UPTODATE 62

SAVAPI is not up to date.

Note:
Not used.

#define SAVAPI3_E_OPTION_NOT_SUPPORTED 23

Option is not supported.

Note:
Trying to set or retrieve a value for an option with an unknown or obsolete id (for instance SAVAPI3_OPTION_UPDATE_SERVERS is obsolete).

#define SAVAPI3_E_OPTION_VALUE_INVALID 76

Invalid option value.

Note:

A configuration command received a value buffer which is not acceptable as a value for the associated option id (for instance it is empty).

#define SAVAPI3_E_PARTIAL   31

Cannot extract multi-volume archive.

Note:

In case of an archive which is part of a multi-volume archive set, a file could not be fully extracted because is split over several archive parts. Processing the next file may be successful if all information is stored in that part.

#define SAVAPI3_E_PARTITION_TABLE   52

Partition tables unequal.

Note:

Not used.

#define SAVAPI3_E_PREFIX_GET   91

Failed to retrieve a detect type option.

Note:

SAVAPI failed to retrieve a detect type option (for instance SAVAPI3_OPTION_DETECT_ADSPY , SAVAPI3_OPTION_DETECT_APPL , others).

#define SAVAPI3_E_PREFIX_SET   90

Failed to set a detect type option.

Note:

SAVAPI failed to set a detect type option (for instance SAVAPI3_OPTION_DETECT_ADSPY , SAVAPI3_OPTION_DETECT_APPL , others)

#define SAVAPI3_E_PROC_ARCHIVE_HANDLE   160

Archive internal handle error.

Note:

An internal handle related to archive processing is invalid or not initialized.

#define SAVAPI3_E_PROC_BAD_FORMAT   154

Bad header format.

Note:
The archive header has been changed with a newer (unsupported) version of a packer application. The archive header is damaged.

#define SAVAPI3_E_PROC_BAD_HEADER   151

Bad archive header.

Note:
The archive header is invalid.

#define SAVAPI3_E_PROC_BAD_TABLE   158

Invalid decompression table.

Note:
Archive contains an invalid decompression table.

#define SAVAPI3_E_PROC_CALLBACK   162

Archive internal callback error.

Note:
Decompression aborted because an internal archive callback is invalid or caused an error.

#define SAVAPI3_E_PROC_DATA_CRC   156

Bad data crc.

Note:
Checksum of compressed data does not match.

#define SAVAPI3_E_PROC_ERROR   29

Archive generic processing error.

Note:
Any other archive scan processing error which is not covered by SAVAPI3_E_PROC_<name> error codes.

#define SAVAPI3_E_PROC_FILE_CRC   157

Bad crc for extracted file.

Note:
Checksum of a decompressed file does not match.

#define SAVAPI3_E_PROC_HEADER_CRC   155

Bad header crc.

Note:
An archive header failed checksum check.

#define SAVAPI3_E_PROC_INCOMPLETE_BLOCK_READ   150

Incomplete archive block read.

Note:
An archive block is damaged and could not be read.

#define SAVAPI3_E_PROC_INVALID_COMPRESSED_DATA   152

Bad compressed data.

Note:
The compressed data from the archive is invalid. Some files could not be extracted and scanned.

#define SAVAPI3_E_PROC_NO_FILES_TO_EXTRACT   161

No files could be extracted.

Note:
Archive is invalid, corrupt or damaged.

#define SAVAPI3_E_PROC_OBSOLETE   153

Obsolete archive information.

Note:
Archive is packed with a very old or a developer version of a packer application and contains obsolete information and unsupported entries.

#define SAVAPI3_E_PROC_TOTAL_LOSS 163

File extraction failed.

Note:
Not all archive contents could be extracted.

#define SAVAPI3_E_PROC_UNEXPECTED_EOF 159

Unexpected end of file.

Note:
Decompression aborted because of unexpected end of file in archive.

#define SAVAPI3_E_QUERY_DISK_PARAM 44

Problem while getting disk geometry.

Note:
Not used.

#define SAVAPI3_E_RECEIVE_FAILED 74

Failed to receive data from the SAVAPI Service.

Note:
The SAVAPI service is not running anymore.
This error can only be triggered by the SAVAPI Client Library.

#define SAVAPI3_E_REPAIR_FAILED 77

Repair an infected file failed.

#define SAVAPI3_E_RESULT_BOOTIMAGE 53

File contains a boot virus image.

Note:
Not used.

#define SAVAPI3_E_RESULT_DROPPER 48

Macro heuristic: possible dropper.

Note:

Not used.

#define SAVAPI3_E_RESULT_ERROR   21

An error occurred during a file scan.

Note:

Not used.

#define SAVAPI3_E_RESULT_FILE_NOT_FOUND   22

Could not extract file.

Note:

A file to extract during an archive scanning could not be found.

#define SAVAPI3_E_RESULT_POLYMORPHIC   50

Macro heuristic: possible polymorphic virus.

Note:

Not used.

#define SAVAPI3_E_RESULT_SUSPICIOUS   35

Possible infected file.

Note:

Not used.

#define SAVAPI3_E_RESULT_TROJAN   49

Macro heuristic: possible trojan horse.

Note:

Not used.

#define SAVAPI3_E_SECTOR_CONSTANT   61

Boot sector contains constant data.

Note:

Not used.

#define SAVAPI3_E_SECTOR_INVALID 39

Invalid sector (no bios signature) (boot record access).

Note:
Not used.

#define SAVAPI3_E_SECTOR_KNOWN 59

Known good boot sector.

Note:
Not used.

#define SAVAPI3_E_SECTOR_READ 37

Read error (boot record access).

Note:
Not used.

#define SAVAPI3_E_SECTOR_UNKNOWN 60

Unknown boot sector.

Note:
Not used.

#define SAVAPI3_E_SECTOR_WRITE 38

Write error (boot record access).

Note:
Not used.

#define SAVAPI3_E_SELFCHK_FILE_CRC 17

Engine file(s) crc check failed.

Note:
One or more engine files failed checksum check because they were manipulated, damaged or truncated.

#define SAVAPI3_E_SELFCHK_FILE_ERR 16

Engine file(s) read failed.

Note:

One or more engine files are damaged or truncated.

#define SAVAPI3_E_SELFCHK_PATCHED 15

Inconsistent versions in engine files set.

Note:

There are incompatible engine files within the engine set which do not match the expected version. The engine file set was not updated or is too old for the present engine set.

#define SAVAPI3_E_SEND_FAILED 75

Failed to send data to the SAVAPI Service.

Note:

The SAVAPI service is not running anymore.
This error can only be triggered by the SAVAPI Client Library.

#define SAVAPI3_E_SETUP_PRODUCT 63

SAVAPI product is not set.

Note:

Not used.

#define SAVAPI3_E_TIMEOUT 34

Scan timed out.

Note:

A scan in progress exceeded the maximum user set scan time-out.

#define SAVAPI3_E_UNKNOWN 81

Unknown engine error.

Note:

Engine returns an unknown error code.

#define SAVAPI3_E_UNSUPPORTED 28

Unsupported archive type/format.

Note:

The archive type is not supported. The version of a known archive type is not supported. The compression method is not supported. The archive format is unknown.

#define SAVAPI3_E_VDF_CRC   10

VDF file(s) crc check failed.

Note:

One ore more VDF files failed checksum check because they were damaged, manipulated or truncated.

#define SAVAPI3_E_VDF_NOT_FOUND   8

VDF file(s) not found.

Note:

Path to the VDF files is not correct, files are missing, or there are no access rights to open the files.

#define SAVAPI3_E_VDF_READ   9

VDF file(s) read failed.

Note:

VDF files are damaged or truncated.

#define SAVAPI3_E_VDF_VERSION   11

Inconsistent versions in VDF files set.

Note:

There are incompatible VDF files within the VDF set. Not all relevant VDF files were downloaded or the engine is too old for the present VDF set.

#define SAVAPI3_E_WRONG_ENGINE   12

Engine initialization failed.

Note:

Engine is too old for this version of SAVAPI. SAVAPI used a wrong character set when initializing the engine.

#define SAVAPI3_E_WRONG_SAVAPI   14

Invalid SAVAPI binary encountered.

Not used.

#define SAVAPI3_S_OK 0

Operation ended with success.

5.3 Error categories

5.3.1 Defines

- #define SAVAPI3_ECAT_ERROR_IO 0
- #define SAVAPI3_ECAT_ERROR_SCAN 1
- #define SAVAPI3_ECAT_ERROR_UNPACK 2
- #define SAVAPI3_ECAT_ERROR_GENERIC 3

5.3.2 Detailed Description

Note:
Used by the error callbacks to categorize the errors they return

5.3.3 Define Documentation

#define SAVAPI3_ECAT_ERROR_GENERIC 3

uncategorised error category

#define SAVAPI3_ECAT_ERROR_IO 0

i/o error category

#define SAVAPI3_ECAT_ERROR_SCAN 1

scan error category

#define SAVAPI3_ECAT_ERROR_UNPACK 2

unpack error category

5.4 Error levels

5.4.1 Defines

- #define SAVAPI3_ELEVEL_ERROR 0
- #define SAVAPI3_ELEVEL_WARNING 1
- #define SAVAPI3_ELEVEL_INFO 2

5.4.2 Detailed Description

Note:
Used by the error callbacks to categorize the returned errors

5.4.3 Define Documentation

#define SAVAPI3_ELEVEL_ERROR 0

error level

#define SAVAPI3_ELEVEL_INFO 2

info level

#define SAVAPI3_ELEVEL_WARNING 1

warning level

5.5 Initialization flags

5.5.1 Defines

- #define SAVAPI3_FLAG_USE_TCP 1
- #define SAVAPI3_FLAG_USE_LOCAL_SOCKET 2

5.5.2 Detailed Description

Note:
Initialization flags will be extended on the fly when needed! The SAVAPI3_FLAG_USE_TCP and SAVAPI3_FLAG_USE_LOCAL_SOCKET must not be set simultaneously

5.5.3 Define Documentation

#define SAVAPI3_FLAG_USE_LOCAL_SOCKET 2

local sockets will be used for communication (SAVAPI client-mode only)

#define SAVAPI3_FLAG_USE_TCP 1

TCP sockets will be used for communication (SAVAPI client-mode only)

5.6 Scan warnings

5.6.1 Defines

- #define SAVAPI3_W_DAMAGED 1
- #define SAVAPI3_W_OLE_DAMAGED 2
- #define SAVAPI3_W_SUSPICIOUS 4
- #define SAVAPI3_W_PROGRESS_ABORT 8
- #define SAVAPI3_W_HEADER_MALFORMED 16
- #define SAVAPI3_W_POTENTIAL_ARCH_BOMB 32
- #define SAVAPI3_W_RATIO_EXCEEDED 64
- #define SAVAPI3_W_MAX_EXTRACTED 128

5.6.2 Detailed Description

Note:
Warnings that can be received during the scanning process

5.6.3 Define Documentation

#define SAVAPI3_W_DAMAGED 1

File has potentially been damaged by virus

#define SAVAPI3_W_HEADER_MALFORMED 16

A malformed archive header was detected

#define SAVAPI3_W_MAX_EXTRACTED 128

Unpacking has reached the maximum limit of extracted data

#define SAVAPI3_W_OLE_DAMAGED 2

OLE-File is potentially damaged

#define SAVAPI3_W_POTENTIAL_ARCH_BOMB 32

This file could be an archive bomb, ratio might be exceeded or something else might have happened to trigger that detection

#define SAVAPI3_W_PROGRESS_ABORT 8

An abort was triggered by the progress callback

#define SAVAPI3_W_RATIO_EXCEEDED 64

The ratio set by the application regarding unpacking size in archives has been exceeded

#define SAVAPI3_W_SUSPICIOUS 4

File is suspicious

5.7 Iframes informations

5.7.1 Defines

- #define SAVAPI3_HTML_CONTENT_ATTRIB_INVISIBLE 1
- #define SAVAPI3_HTML_CONTENT_ATTRIB_EXTRASMALL 2
- #define SAVAPI3_HTML_CONTENT_ATTRIB_ODDPOS 4
- #define SAVAPI3_HTML_CONTENT_ATTRIB_MALICIOUS 8

5.7.2 Detailed Description

Note:

Informations that can be received during the scanning process

---

5.7.3 Define Documentation

#define SAVAPI3_HTML_CONTENT_ATTRIB_EXTRASMALL 2

The object is very small and as such almost invisible to the user surfing the site

#define SAVAPI3_HTML_CONTENT_ATTRIB_INVISIBLE 1

The object is invisible to the user surfing the site

#define SAVAPI3_HTML_CONTENT_ATTRIB_MALICIOUS 8

The object is likely of a malicious nature

#define SAVAPI3_HTML_CONTENT_ATTRIB_ODDPOS 4

The object is inserted at a very uncommon position in the HTML code

5.8 Scan informations

5.8.1 Defines

- #define SAVAPI3_I_OLEFILE 1
- #define SAVAPI3_I_TEMPLATE 2
- #define SAVAPI3_I_MACROS_PRESENT 4
- #define SAVAPI3_I_ALL_MACROS_DELETED 8
- #define SAVAPI3_I_OLE_ENCRYPTED 16
- #define SAVAPI3_I_ACTIVE_CONTENT_PRESENT 32
- #define SAVAPI3_I_MAILBOX 64

5.8.2 Detailed Description

Note:
Informations that can be received during the scanning process

5.8.3 Define Documentation

#define SAVAPI3_I_ACTIVE_CONTENT_PRESENT 32

SCRIPT: html contains active content (JS/VBS, etc.)

#define SAVAPI3_I_ALL_MACROS_DELETED 8

OLE: all macros were deleted

#define SAVAPI3_I_MACROS_PRESENT 4

OLE: contains macros

#define SAVAPI3_I_MAILBOX 64

ARCHIVE: Mailbox detected

#define SAVAPI3_I_OLE_ENCRYPTED 16

OLE: encrypted marker, only for DOCs

#define SAVAPI3_I_OLEFILE 1

OLE: file is a compound doc (OLE2)

#define SAVAPI3_I_TEMPLATE 2

OLE: contains a word template

5.9 SAVAPI options

5.9.1 Modules

- GET/SET options (read/write)

  "SET" requests are available to configure SAVAPI. For the following requests, a "GET" counterpart is also available and these are therefore labeled as "read/write". Only the "SET" version is listed here although a "GET" version also exists. The "GET" response will return the same data that is provided with the "SET" request (although the representation of the data may be different. For example, a "SET" request with "10K" could lead to a "GET" response with "10240".)

- SET options (write only)

"SET" requests are available to configure SAVAPI. Usually a "SET" request also has a "GET" request counterpart to retrieve current settings. However, the following commands do not have a "GET" counterpart and are therefore labeled as "write only".

- GET options (read only)

  "GET" requests are available to retrieve current SAVAPI settings. Usually a "GET" request also has a "SET" request counterpart to configure SAVAPI. However, the following commands do not have a "SET" counterpart and are therefore labeled as "read only".

5.9.2 Detailed Description

Remarks:

Almost each option used to configure the SAVAPI instance (the paths to the temporary folders, the scanning options) has a default value that is written in its description as a note (i.e. Default value: <value>).
In client-mode, the default values are dependent to the configuration used to start the SAVAPI service, so the provided defaults only applies in library-mode!!!
The options that has no default (i.e. unsupported options, obsolete, ignored) will be marked with the "Default value: None" string.

5.10 GET/SET options (read/write)

"SET" requests are available to configure SAVAPI. For the following requests, a "GET" counterpart is also available and these are therefore labeled as "read/write". Only the "SET" version is listed here although a "GET" version also exists. The "GET" response will return the same data that is provided with the "SET" request (although the representation of the data may be different. For example, a "SET" request with "10K" could lead to a "GET" response with "10240".)

5.10.1 Defines

- #define SAVAPI3_PUBLIC_OPTIONS   0
  Marks the start of the space used for SAVAPI public options.
- #define SAVAPI3_OPTION_CWD   SAVAPI3_PUBLIC_OPTIONS + 1
  Specifies current working directory for SAVAPI.
- #define SAVAPI3_OPTION_CONF   SAVAPI3_PUBLIC_OPTIONS + 2
  Specifies the configuration file that is used.
- #define SAVAPI3_OPTION_ARCHIVE_SCAN   SAVAPI3_PUBLIC_OPTIONS + 3
  Activates archive detection and scanning.
- #define SAVAPI3_OPTION_ARCHIVE_MAX_SIZE   SAVAPI3_PUBLIC_OPTIONS + 4
  Set the maximum allowed size (in bytes) for any file within an archive.
- #define SAVAPI3_OPTION_ARCHIVE_MAX_REC   SAVAPI3_PUBLIC_OPTIONS + 5
  Set the maximum allowed recursion within an archive.
- #define SAVAPI3_OPTION_ARCHIVE_MAX_RATIO   SAVAPI3_PUBLIC_OPTIONS + 6
  Set the maximum allowed decompressing-ratio within an archive.
- #define SAVAPI3_OPTION_ARCHIVE_MAX_COUNT   SAVAPI3_PUBLIC_OPTIONS + 7
  Set the maximum allowed number of files within an archive.
- #define SAVAPI3_OPTION_MAILBOX_SCAN   SAVAPI3_PUBLIC_OPTIONS + 8
  Activates detection and scanning of mailboxes.

- #define SAVAPI3_OPTION_HEUR_MACRO SAVAPI3_PUBLIC_OPTIONS + 9
  Activates heuristic macro detection.
- #define SAVAPI3_OPTION_HEUR_LEVEL SAVAPI3_PUBLIC_OPTIONS + 10
  Set the heuristic level for the engine. The available levels are:
  - 0 - Disable heuristic detection.
  - 1 - Lazy heuristic detection. This is the lowest possible mode, detection is not very good, but the false positives number will be low.
  - 2 - Normal heuristic detection.
  - 3 - High heuristic detection. This is the highest possible mode, but the false positives number will be high.

- #define SAVAPI3_OPTION_SCAN_TEMP SAVAPI3_PUBLIC_OPTIONS + 11
  Set the temporary directory used for scanning files.
- #define SAVAPI3_OPTION_SCAN_TIMEOUT SAVAPI3_PUBLIC_OPTIONS + 12
  Set the maximum number of seconds allowed to scan a file before aborting.
- #define SAVAPI3_OPTION_REPAIR SAVAPI3_PUBLIC_OPTIONS + 13
  Activates the repairing of infected files.
- #define SAVAPI3_OPTION_NOTIFY_REPAIR SAVAPI3_PUBLIC_OPTIONS + 14
  Activates the notification of reparable infected files.
- #define SAVAPI3_OPTION_NOTIFY_OFFICE SAVAPI3_PUBLIC_OPTIONS + 15
  Activates the detection of office documents.
- #define SAVAPI3_OPTION_NOTIFY_OFFICE_MACRO SAVAPI3_PUBLIC_OPTIONS + 16
  Activates the detection of macros within office documents.
- #define SAVAPI3_OPTION_UPDATE_SERVERS SAVAPI3_PUBLIC_OPTIONS + 17
  Specify the list of update servers.
- #define SAVAPI3_OPTION_UPDATE_PROXY SAVAPI3_PUBLIC_OPTIONS + 18
  Activate usage of a proxy server for updates.
- #define SAVAPI3_OPTION_UPDATE_PROXY_SETTINGS SAVAPI3_PUBLIC_OPTIONS + 19
  Specify the settings for a proxy server.
- #define SAVAPI3_OPTION_NOTIFY_ALERTURL SAVAPI3_PUBLIC_OPTIONS + 20
  Activates the notification of virus description url.
- #define SAVAPI3_OPTION_DETECT_ADSPY SAVAPI3_PUBLIC_OPTIONS + 21
  Activate detection for the specified type.
- #define SAVAPI3_OPTION_DETECT_APPL SAVAPI3_PUBLIC_OPTIONS + 22
  Activate detection for the specified type.
- #define SAVAPI3_OPTION_DETECT_BDC SAVAPI3_PUBLIC_OPTIONS + 23
  Activate detection for the specified type.
- #define SAVAPI3_OPTION_DETECT_DIAL SAVAPI3_PUBLIC_OPTIONS + 24
  Activate detection for the specified type.
- #define SAVAPI3_OPTION_DETECT_GAME SAVAPI3_PUBLIC_OPTIONS + 25
  Activate detection for the specified type.
- #define SAVAPI3_OPTION_DETECT_HIDDENEXT SAVAPI3_PUBLIC_OPTIONS + 26
  Activate detection for the specified type.
- #define SAVAPI3_OPTION_DETECT_JOKE SAVAPI3_PUBLIC_OPTIONS + 27
  Activate detection for the specified type.
- #define SAVAPI3_OPTION_DETECT_PCK SAVAPI3_PUBLIC_OPTIONS + 28
  Activate detection for the specified type.
- #define SAVAPI3_OPTION_DETECT_PHISH SAVAPI3_PUBLIC_OPTIONS + 29

Activate detection for the specified type.

- #define SAVAPI3_OPTION_DETECT_SPR   SAVAPI3_PUBLIC_OPTIONS + 30
  Activate detection for the specified type.
- #define SAVAPI3_OPTION_IFRAMES_URL   SAVAPI3_PUBLIC_OPTIONS + 31
  Activate IFRAME detection.
- #define SAVAPI3_OPTION_REPORT_ENCRYPTED_MIME   SAVAPI3_PUBLIC_OPTIONS + 32
  Activate reporting of encrypted mails.
- #define SAVAPI3_OPTION_SCAN_MODE   SAVAPI3_PUBLIC_OPTIONS + 33
  Set the scanning method. Available options are:
  - SMART - Smart Extensions scan mode. The files scanned for malware are chosen by SAVAPI The choice is made based on the files content. This is the recommended setting.
  - ALL - All scan mode. Files are scanned for malware, no matter their content or extension.
  - EXTLIST - Extensions List scan mode. Only files with specific extensions are scanned for malware content.
- #define SAVAPI3_OPTION_MIME_SCAN   SAVAPI3_PUBLIC_OPTIONS + 34
  Activate detection and scanning of mails.
- #define SAVAPI3_OPTION_PGP_SCAN   SAVAPI3_PUBLIC_OPTIONS + 35
  Activate scanning and reporting of PGP binaries.

5.10.2 Detailed Description

"SET" requests are available to configure SAVAPI. For the following requests, a "GET" counterpart is also available and these are therefore labeled as "read/write". Only the "SET" version is listed here although a "GET" version also exists. The "GET" response will return the same data that is provided with the "SET" request (although the representation of the data may be different. For example, a "SET" request with "10K" could lead to a "GET" response with "10240".)

5.10.3 Define Documentation

#define SAVAPI3_OPTION_ARCHIVE_MAX_COUNT   SAVAPI3_PUBLIC_OPTIONS + 7

Set the maximum allowed number of files within an archive.

Note:

A value of "0" means the maximum allowed value (INT64_MAX).
This setting has no meaning if ARCHIVE_SCAN is deactivated.
Default value: 0

#define SAVAPI3_OPTION_ARCHIVE_MAX_RATIO   SAVAPI3_PUBLIC_OPTIONS + 6

Set the maximum allowed decompressing-ratio within an archive.

Note:

A value of "0" means the maximum allowed value (INT32_MAX).
This setting has no meaning if ARCHIVE_SCAN is deactivated.
Default value: 150

#define SAVAPI3_OPTION_ARCHIVE_MAX_REC   SAVAPI3_PUBLIC_OPTIONS + 5

Set the maximum allowed recursion within an archive.

Note:

A value of "0" means the maximum allowed value (1000 recursion levels).
This setting has no meaning if ARCHIVE_SCAN is deactivated.
Default value: 200

#define SAVAPI3_OPTION_ARCHIVE_MAX_SIZE   SAVAPI3_PUBLIC_OPTIONS + 4

Set the maximum allowed size (in bytes) for any file within an archive.

Note:

A value of "0" means the maximum allowed value (INT64_MAX bytes).
This setting has no meaning if ARCHIVE_SCAN is deactivated.
Default value: 1073741824 (1G)

#define SAVAPI3_OPTION_ARCHIVE_SCAN   SAVAPI3_PUBLIC_OPTIONS + 3

Activates archive detection and scanning.

Note:

Default value: 0 (disabled)

#define SAVAPI3_OPTION_CONF   SAVAPI3_PUBLIC_OPTIONS + 2

Specifies the configuration file that is used.

Note:

The configuration file will be (re-)read as part of this request.
Available only in client-mode.
Default value: None

#define SAVAPI3_OPTION_CWD   SAVAPI3_PUBLIC_OPTIONS + 1

Specifies current working directory for SAVAPI.

Note:

This eliminates the need to specify full paths in filenames.
Available only in client-mode.
Default value: None

#define SAVAPI3_OPTION_DETECT_ADSPY   SAVAPI3_PUBLIC_OPTIONS + 21

Activate detection for the specified type.

Note:

Default value: 1 (enabled)

#define SAVAPI3_OPTION_DETECT_APPL   SAVAPI3_PUBLIC_OPTIONS + 22

Activate detection for the specified type.

Note:

Default value: 0 (disabled)

#define SAVAPI3_OPTION_DETECT_BDC   SAVAPI3_PUBLIC_OPTIONS + 23

Activate detection for the specified type.

Note:

Default value: 1 (enabled)

#define SAVAPI3_OPTION_DETECT_DIAL   SAVAPI3_PUBLIC_OPTIONS + 24

Activate detection for the specified type.

Note:

Default value: 1 (enabled)

#define SAVAPI3_OPTION_DETECT_GAME   SAVAPI3_PUBLIC_OPTIONS + 25

Activate detection for the specified type.

Note:

Default value: 0 (disabled)

#define SAVAPI3_OPTION_DETECT_HIDDENEXT   SAVAPI3_PUBLIC_OPTIONS + 26

Activate detection for the specified type.

Note:

Default value: 1 (enabled)

#define SAVAPI3_OPTION_DETECT_JOKE   SAVAPI3_PUBLIC_OPTIONS + 27

Activate detection for the specified type.

Note:

Default value: 0 (disabled)

#define SAVAPI3_OPTION_DETECT_PCK   SAVAPI3_PUBLIC_OPTIONS + 28

Activate detection for the specified type.

Note:

Default value: 0 (disabled)

#define SAVAPI3_OPTION_DETECT_PHISH   SAVAPI3_PUBLIC_OPTIONS + 29

Activate detection for the specified type.

Note:

Default value: 1 (enabled)

#define SAVAPI3_OPTION_DETECT_SPR   SAVAPI3_PUBLIC_OPTIONS + 30

Activate detection for the specified type.

Note:

Default value: 0 (disabled)

#define SAVAPI3_OPTION_HEUR_LEVEL   SAVAPI3_PUBLIC_OPTIONS + 10

Set the heuristic level for the engine. The available levels are:

- 0 - Disable heuristic detection.
- 1 - Lazy heuristic detection. This is the lowest possible mode, detection is not very good, but the false positives number will be low.
- 2 - Normal heuristic detection.
- 3 - High heuristic detection. This is the highest possible mode, but the false positives number will be high.

Note:

Default value: 0 (disabled)

#define SAVAPI3_OPTION_HEUR_MACRO   SAVAPI3_PUBLIC_OPTIONS + 9

Activates heuristic macro detection.

Note:

Default value: 1 (enabled)

#define SAVAPI3_OPTION_IFRAMES_URL SAVAPI3_PUBLIC_OPTIONS + 31

Activate IFRAME detection.

Note:
Default value: 0 (disabled)

#define SAVAPI3_OPTION_MAILBOX_SCAN SAVAPI3_PUBLIC_OPTIONS + 8

Activates detection and scanning of mailboxes.

Note:
Default value: 0 (disabled)

#define SAVAPI3_OPTION_MIME_SCAN SAVAPI3_PUBLIC_OPTIONS + 34

Activate detection and scanning of mails.

Note:
Default value: 1 (enabled)

#define SAVAPI3_OPTION_NOTIFY_ALERTURL SAVAPI3_PUBLIC_OPTIONS + 20

Activates the notification of virus description url.

Note:
Default value: 0 (disabled)

#define SAVAPI3_OPTION_NOTIFY_OFFICE SAVAPI3_PUBLIC_OPTIONS + 15

Activates the detection of office documents.

Note:
Default value: 0 (disabled)

#define SAVAPI3_OPTION_NOTIFY_OFFICE_MACRO SAVAPI3_PUBLIC_OPTIONS + 16

Activates the detection of macros within office documents.

Note:
Default value: 0 (disabled)

#define SAVAPI3_OPTION_NOTIFY_REPAIR SAVAPI3_PUBLIC_OPTIONS + 14

Activates the notification of reparable infected files.

Note:

Default value: 0 (disabled)

#define SAVAPI3_OPTION_PGP_SCAN   SAVAPI3_PUBLIC_OPTIONS + 35

Activate scanning and reporting of PGP binaries.

Note:

Default value: 0 (disabled)

#define SAVAPI3_OPTION_REPAIR   SAVAPI3_PUBLIC_OPTIONS + 13

Activates the repairing of infected files.

Note:

Default value: 0 (disabled)

#define SAVAPI3_OPTION_REPORT_ENCRYPTED_MIME   SAVAPI3_PUBLIC_OPTIONS + 32

Activate reporting of encrypted mails.

Note:

Default value: 0 (disabled)

#define SAVAPI3_OPTION_SCAN_MODE   SAVAPI3_PUBLIC_OPTIONS + 33

Set the scanning method. Available options are:

- SMART - Smart Extensions scan mode. The files scanned for malware are chosen by SAVAPI The choice is made based on the files content. This is the recommended setting.
- ALL - All scan mode. Files are scanned for malware, no matter their content or extension.
- EXTLIST - Extensions List scan mode. Only files with specific extensions are scanned for malware content.

Note:

Default value: SMART

#define SAVAPI3_OPTION_SCAN_TEMP   SAVAPI3_PUBLIC_OPTIONS + 11

Set the temporary directory used for scanning files.

Note:

SAVAPI may use other temporary directories for files that are not being scanned. These other directories can be specified with command-line arguments or in a configuration file.

Default value: The system temporary folder

#define SAVAPI3_OPTION_SCAN_TIMEOUT   SAVAPI3_PUBLIC_OPTIONS + 12

Set the maximum number of seconds allowed to scan a file before aborting.

Note:
Default value: 0 (no timeout)

#define SAVAPI3_OPTION_UPDATE_PROXY   SAVAPI3_PUBLIC_OPTIONS + 18

Activate usage of a proxy server for updates.

Note:
This option is obsolete and is kept only for backward compatibility.
Trying to use this option will result in SAVAPI3_E_OPTION_NOT_SUPPORTED error.
Default value: None

#define SAVAPI3_OPTION_UPDATE_PROXY_SETTINGS   SAVAPI3_PUBLIC_OPTIONS + 19

Specify the settings for a proxy server.

Note:
This option is obsolete and is kept only for backward compatibility.
Trying to use this option will result in SAVAPI3_E_OPTION_NOT_SUPPORTED error.
Default value: None

#define SAVAPI3_OPTION_UPDATE_SERVERS   SAVAPI3_PUBLIC_OPTIONS + 17

Specify the list of update servers.

Note:
This option is obsolete and is kept only for backward compatibility.
Trying to use this option will result in SAVAPI3_E_OPTION_NOT_SUPPORTED error.
Default value: None

#define SAVAPI3_PUBLIC_OPTIONS   0

Marks the start of the space used for SAVAPI public options.

5.11 SET options (write only)

"SET" requests are available to configure SAVAPI. Usually a "SET" request also has a "GET" request counterpart to retrieve current settings. However, the following commands do not have a "GET" counterpart and are therefore labeled as "write only".

5.11.1 Defines

- #define SAVAPI3_OPTION_PRODUCT SAVAPI3_PUBLIC_OPTIONS + 40
  Set the key-id that is required by the application.
- #define SAVAPI3_OPTION_DETECT_ALLTYPES SAVAPI3_PUBLIC_OPTIONS + 41

5.11.2 Detailed Description

"SET" requests are available to configure SAVAPI. Usually a "SET" request also has a "GET" request counterpart to retrieve current settings. However, the following commands do not have a "GET" counterpart and are therefore labeled as "write only".

5.11.3 Define Documentation

#define SAVAPI3_OPTION_DETECT_ALLTYPES SAVAPI3_PUBLIC_OPTIONS + 41

Activate detection for all types.

#define SAVAPI3_OPTION_PRODUCT SAVAPI3_PUBLIC_OPTIONS + 40

Set the key-id that is required by the application.

Note:

SAVAPI will check if the key-id is within the license and that it is not expired. If it is available and is valid, the application is free to use SAVAPI. If not, most requests will result in an error response.

5.12 GET options (read only)

"GET" requests are available to retrieve current SAVAPI settings. Usually a "GET" request also has a "SET" request counterpart to configure SAVAPI. However, the following commands do not have a "SET" counterpart and are therefore labeled as "read only".

5.12.1 Defines

- #define SAVAPI3_OPTION_SAVAPI SAVAPI3_PUBLIC_OPTIONS + 50
- #define SAVAPI3_OPTION_AVE_VERSION SAVAPI3_PUBLIC_OPTIONS + 51
- #define SAVAPI3_OPTION_VDF_VERSION SAVAPI3_PUBLIC_OPTIONS + 52
- #define SAVAPI3_OPTION_PID SAVAPI3_PUBLIC_OPTIONS + 53
  Retrieve the process-id for the SAVAPI process that is currently handling the TCP/IP connection.
- #define SAVAPI3_OPTION_EXPIRE SAVAPI3_PUBLIC_OPTIONS + 54
- #define SAVAPI3_OPTION_VDFSIGCOUNT SAVAPI3_PUBLIC_OPTIONS + 55
- #define SAVAPI3_OPTION_SELECTABLE_DETECT SAVAPI3_PUBLIC_OPTIONS + 56
  Retrieve the various types that can be detected (and dynamically turned on/off).
- #define SAVAPI3_OPTION_DESCR_ADSPY SAVAPI3_PUBLIC_OPTIONS + 57
- #define SAVAPI3_OPTION_DESCR_APPL SAVAPI3_PUBLIC_OPTIONS + 58
- #define SAVAPI3_OPTION_DESCR_BDC SAVAPI3_PUBLIC_OPTIONS + 59
- #define SAVAPI3_OPTION_DESCR_DIAL SAVAPI3_PUBLIC_OPTIONS + 60

- #define SAVAPI3_OPTION_DESCR_GAME SAVAPI3_PUBLIC_OPTIONS + 61
- #define SAVAPI3_OPTION_DESCR_HIDDENEXT SAVAPI3_PUBLIC_OPTIONS + 62
- #define SAVAPI3_OPTION_DESCR_JOKE SAVAPI3_PUBLIC_OPTIONS + 63
- #define SAVAPI3_OPTION_DESCR_PCK SAVAPI3_PUBLIC_OPTIONS + 64
- #define SAVAPI3_OPTION_DESCR_PHISH SAVAPI3_PUBLIC_OPTIONS + 65
- #define SAVAPI3_OPTION_DESCR_SPR SAVAPI3_PUBLIC_OPTIONS + 66
- #define SAVAPI3_OPTION_VDF_DATE SAVAPI3_PUBLIC_OPTIONS + 67

5.12.2 Detailed Description

"GET" requests are available to retrieve current SAVAPI settings. Usually a "GET" request also has a "SET" request counterpart to configure SAVAPI. However, the following commands do not have a "SET" counterpart and are therefore labeled as "read only".

5.12.3 Define Documentation

#define SAVAPI3_OPTION_AVE_VERSION SAVAPI3_PUBLIC_OPTIONS + 51

Retrieve engine version number

#define SAVAPI3_OPTION_DESCR_ADSPY SAVAPI3_PUBLIC_OPTIONS + 57

Retrieve the english description for the given type.

#define SAVAPI3_OPTION_DESCR_APPL SAVAPI3_PUBLIC_OPTIONS + 58

Retrieve the english description for the given type.

#define SAVAPI3_OPTION_DESCR_BDC SAVAPI3_PUBLIC_OPTIONS + 59

Retrieve the english description for the given type.

#define SAVAPI3_OPTION_DESCR_DIAL SAVAPI3_PUBLIC_OPTIONS + 60

Retrieve the english description for the given type.

#define SAVAPI3_OPTION_DESCR_GAME SAVAPI3_PUBLIC_OPTIONS + 61

Retrieve the english description for the given type.

#define SAVAPI3_OPTION_DESCR_HIDDENEXT SAVAPI3_PUBLIC_OPTIONS + 62

Retrieve the english description for the given type.

#define SAVAPI3_OPTION_DESCR_JOKE SAVAPI3_PUBLIC_OPTIONS + 63

Retrieve the english description for the given type.

#define SAVAPI3_OPTION_DESCR_PCK SAVAPI3_PUBLIC_OPTIONS + 64

Retrieve the english description for the given type.

#define SAVAPI3_OPTION_DESCR_PHISH SAVAPI3_PUBLIC_OPTIONS + 65

Retrieve the english description for the given type.

#define SAVAPI3_OPTION_DESCR_SPR SAVAPI3_PUBLIC_OPTIONS + 66

Retrieve the english description for the given type.

#define SAVAPI3_OPTION_EXPIRE SAVAPI3_PUBLIC_OPTIONS + 54

Retrieve the expiration date of the SAVAPI license (YYYYMMDD)

#define SAVAPI3_OPTION_PID SAVAPI3_PUBLIC_OPTIONS + 53

Retrieve the process-id for the SAVAPI process that is currently handling the TCP/IP connection.

Note:

Available only in client-mode.

#define SAVAPI3_OPTION_SAVAPI SAVAPI3_PUBLIC_OPTIONS + 50

Retrieve SAVAPI protocol version number

#define SAVAPI3_OPTION_SELECTABLE_DETECT SAVAPI3_PUBLIC_OPTIONS + 56

Retrieve the various types that can be detected (and dynamically turned on/off).

Note:

The types are returned as a comma separated list. The current value (Jan 2006) would be:
ADSPY,BDC,DIAL,GAME,HEUR_DBLEXT,JOKE,PCK,SPR

#define SAVAPI3_OPTION_VDF_DATE SAVAPI3_PUBLIC_OPTIONS + 67

Retrieve the license date of the vdf(-set). Date is the form of YYYYMMDD.

#define SAVAPI3_OPTION_VDF_VERSION SAVAPI3_PUBLIC_OPTIONS + 52

Retrieve vdf(-set) version number

#define SAVAPI3_OPTION_VDFSIGCOUNT SAVAPI3_PUBLIC_OPTIONS + 55

Retrieve the number of signatures in the vdf(-set)

5.13 Callbacks' ids

5.13.1 Defines

- #define SAVAPI3_CALLBACK_REPORT_FILE_STATUS 0
  Triggered after a file is scanned. The callback data contains the status of the last scanned file.
- #define SAVAPI3_CALLBACK_REPORT_ERROR 3
  Triggered to report an error or a warning.
- #define SAVAPI3_CALLBACK_PRE_SCAN 4
  Triggered before the scanning begins. Can be used to create filters. For example, if we want to scan only .exe files, we install a PRE_SCAN callback. Before each file is scanned, the PRE_SCAN callback will be called.

Inside our implementation of the callback, we implement the filter. If the returned code is success, the file will be scanned, otherwise it will be skipped.

- #define SAVAPI3_CALLBACK_ARCHIVE_OPEN 5
  Triggered before opening an archive. If the returned code is success, the archive will be opened, otherwise it will be skipped from opening.
- #define SAVAPI3_CALLBACK_PROGRESS_REPORT 6
  Triggered when messages related to scan progress are available.
- #define SAVAPI3_CALLBACK_CONTENT_REPORT 7
  Triggered when messages related to scan (progress, warnings or infos that are not error_callback related) are available. IFRAME detection (SAVAPI3_OPTION_IFRAMES_URL) will be reported through this callback.
- #define SAVAPI3_CALLBACK_SCAN_DETAILS_REPORT 8
  Triggered when messages related to scan process details are available. The virus description url (SAVAPI3_OPTION_NOTIFY_ALERTURL).

5.13.2 Define Documentation

#define SAVAPI3_CALLBACK_ARCHIVE_OPEN 5

Triggered before opening an archive. If the returned code is success, the archive will be opened, otherwise it will be skipped from opening.

Note:
Currently, this callback is not triggered in client-mode.

#define SAVAPI3_CALLBACK_CONTENT_REPORT 7

Triggered when messages related to scan (progress, warnings or infos that are not error_callback related) are available. IFRAME detection (SAVAPI3_OPTION_IFRAMES_URL) will be reported through this callback.

#define SAVAPI3_CALLBACK_PRE_SCAN 4

Triggered before the scanning begins. Can be used to create filters. For example, if we want to scan only .exe files, we install a PRE_SCAN callback. Before each file is scanned, the PRE_SCAN callback will be called. Inside our implementation of the callback, we implement the filter. If the returned code is success, the file will be scanned, otherwise it will be skipped.

Note:
Currently, this callback is not triggered in client-mode.

#define SAVAPI3_CALLBACK_PROGRESS_REPORT 6

Triggered when messages related to scan progress are available.

#define SAVAPI3_CALLBACK_REPORT_ERROR 3

Triggered to report an error or a warning.

Note:

Can be triggered at any time.

#define SAVAPI3_CALLBACK_REPORT_FILE_STATUS 0

Triggered after a file is scanned. The callback data contains the status of the last scanned file.

#define SAVAPI3_CALLBACK_SCAN_DETAILS_REPORT 8

Triggered when messages related to scan process details are available. The virus description url (SAVAPI3_OPTION_NOTIFY_ALERTURL).

5.14 SAVAPI report scan details types

5.14.1 Defines

- #define SAVAPI3_REPORT_ALERTURL 1
- #define SAVAPI3_REPORT_REPAIRABLE 2

5.14.2 Define Documentation

#define SAVAPI3_REPORT_ALERTURL 1

Malware URL description

#define SAVAPI3_REPORT_REPAIRABLE 2

Malware found in the object can be repaired

5.15 SAVAPI report content types

5.15.1 Defines

- #define SAVAPI3_REPORT_CONTENT_IFRAME 0

5.15.2 Define Documentation

#define SAVAPI3_REPORT_CONTENT_IFRAME 0

IFRAME URL report

5.16 SAVAPI signals

5.16.1 Defines

- #define SAVAPI3_SIGNAL_SCAN_ABORT 1
  Will cause the SAVAPI instance to abort scanning process as soon as possible.

5.16.2 Define Documentation

#define SAVAPI3_SIGNAL_SCAN_ABORT 1

Will cause the SAVAPI instance to abort scanning process as soon as possible.

Note:
The signal have no associated specific data. When calling SAVAPI3_send_signal function, "data" argument may be NULL.

Todo:
Add new signals as needed.

5.17 SAVAPI commands

5.17.1 Defines

- #define SAVAPI3_COMMAND_RELOAD 0
  Synchronous commands for the SAVAPI client-mode.
- #define SAVAPI3_COMMAND_SHUTDOWN 2
- #define SAVAPI3_COMMAND_PING 3
- #define SAVAPI3_COMMAND_UPDATE_CHECK 4
- #define SAVAPI3_COMMAND_UPDATE_START 5

5.17.2 Define Documentation

#define SAVAPI3_COMMAND_PING 3

Test if SAVAPI daemon is alive

#define SAVAPI3_COMMAND_RELOAD 0

Synchronous commands for the SAVAPI client-mode.

Reload SAVAPI original configuration files

#define SAVAPI3_COMMAND_SHUTDOWN 2

Shutdown SAVAPI

#define SAVAPI3_COMMAND_UPDATE_CHECK 4

Check for updates

#define SAVAPI3_COMMAND_UPDATE_START 5

Perform the update if available

5.18 SAVAPI scan statuses

5.18.1 Defines

- #define SAVAPI3_SCAN_STATUS_CLEAN 0
- #define SAVAPI3_SCAN_STATUS_INFECTED 1
- #define SAVAPI3_SCAN_STATUS_SUSPICIOUS 2
- #define SAVAPI3_SCAN_STATUS_ERROR 3
- #define SAVAPI3_SCAN_STATUS_FINISHED 4

5.18.2 Define Documentation

#define SAVAPI3_SCAN_STATUS_CLEAN 0

Processed object is clean.

#define SAVAPI3_SCAN_STATUS_ERROR 3

An error occurred during object processing

#define SAVAPI3_SCAN_STATUS_FINISHED 4

Object processing finished.

#define SAVAPI3_SCAN_STATUS_INFECTED 1

Viral code found during object processing.

#define SAVAPI3_SCAN_STATUS_SUSPICIOUS 2

Suspicious code found during object processing.

5.19 File types

5.19.1 Defines

- #define SAVAPI3_FTYPE_REGULAR 4
- #define SAVAPI3_FTYPE_ARCHIVE 1
- #define SAVAPI3_FTYPE_IN_ARCHIVE 2

5.19.2 Detailed Description

Note:
Used by the callbacks to report the type of the scanned file

5.19.3 Define Documentation

#define SAVAPI3_FTYPE_ARCHIVE 1

Known archive type

#define SAVAPI3_FTYPE_IN_ARCHIVE 2

File is in an archive

#define SAVAPI3_FTYPE_REGULAR 4

Regular file (all files are regular)

5.20 SAVAPI structures

5.20.1 Data Structures

- struct SAVAPI3_global_init
- *The structure used at SAVAPI initialization.* struct SAVAPI3_instance_init
- *The structure used at SAVAPI instance creation.* struct SAVAPI3_file_info
- *Contains data about the scanned file.* struct SAVAPI3_malware_info
- *Contains data about the found malware in an infected/suspicious file.* struct SAVAPI3_pre_scan_data
- *Contains the data sent to a prescan callback.* struct SAVAPI3_archive_open_data
- *Contains the data sent to a archive_open callback.* struct SAVAPI3_key_value
- *Generic container.* struct SAVAPI3_file_status_data
- *Contains the data sent to a report file status callback.* struct SAVAPI3_error_data
- *The structure associated with report error callback.* struct SAVAPI3_report_progress_data
- *The structure associated with report progress callback.* struct SAVAPI3_iframe_url_data
- *Structure associated with the iframe report.* struct SAVAPI3_report_content_data
- *The structure associated with report content callback.* struct SAVAPI3_alert_url_data
- *Structure associated with the ALERTURL report.* struct SAVAPI3_repairable_data
- *Structure associated with the REPAIRABLE report.* struct SAVAPI3_report_scan_details_data
- *The structure associated with report scan details callback.* struct SAVAPI3_callback_data
- *Structure passed by SAVAPI to a user defined callback, containing all the necessary data.* struct SAVAPI3_signal_data
- *The structure to be passed when sending a signal.* struct SAVAPI3_command_data
- *The structure to be passed when sending a command.* struct SAVAPI3_version

5.20.2 *The structure used to retrieve SAVAPI version.* Typedefs

- typedef struct SAVAPI3_global_init SAVAPI3_GLOBAL_INIT
  The structure used at SAVAPI initialization.
- typedef struct SAVAPI3_instance_init SAVAPI3_INSTANCE_INIT
  The structure used at SAVAPI instance creation.

- typedef struct SAVAPI3_file_info SAVAPI3_FILE_INFO
  Contains data about the scanned file.
- typedef struct SAVAPI3_malware_info SAVAPI3_MALWARE_INFO
  Contains data about the found malware in an infected/suspicious file.
- typedef struct SAVAPI3_pre_scan_data SAVAPI3_PRESCAN_DATA
  Contains the data sent to a prescan callback.
- typedef struct SAVAPI3_archive_open_data SAVAPI3_ARCHIVE_OPEN_DATA
  Contains the data sent to a archive_open callback.
- typedef struct SAVAPI3_key_value SAVAPI3_KEY_VALUE
  Generic container.
- typedef struct SAVAPI3_file_status_data SAVAPI3_FILE_STATUS_DATA
  Contains the data sent to a report file status callback.
- typedef struct SAVAPI3_error_data SAVAPI3_ERROR_DATA
  The structure associated with report error callback.
- typedef struct SAVAPI3_report_progress_data SAVAPI3_REPORT_PROGRESS_DATA
  The structure associated with report progress callback.
- typedef struct SAVAPI3_iframe_url_data SAVAPI3_IFRAME_URL_DATA
  Structure associated with the iframe report.
- typedef struct SAVAPI3_report_content_data SAVAPI3_REPORT_CONTENT_DATA
  The structure associated with report content callback.
- typedef struct SAVAPI3_alert_url_data SAVAPI3_ALERT_URL_DATA
  Structure associated with the ALERTURL report.
- typedef struct SAVAPI3_repairable_data SAVAPI3_REPAIRABLE_DATA
  Structure associated with the REPAIRABLE report.
- typedef struct SAVAPI3_report_scan_details_data SAVAPI3_REPORT_SCAN_DETAILS_DATA
  The structure associated with report scan details callback.
- typedef struct SAVAPI3_callback_data SAVAPI3_CALLBACK_DATA
  Structure passed by SAVAPI to a user defined callback, containing all the necessary data.
- typedef struct SAVAPI3_signal_data SAVAPI3_SIGNAL_DATA
  The structure to be passed when sending a signal.
- typedef struct SAVAPI3_command_data SAVAPI3_COMMAND_DATA
  The structure to be passed when sending a command.
- typedef struct SAVAPI3_version SAVAPI3_VERSION
  The structure used to retrieve SAVAPI version.
- typedef enum _SAVAPI3_log_level SAVAPI3_LOG_LEVEL
  The enumeration used to specify the SAVAPI's logging levels.

5.20.3 Enumerations

- enum _SAVAPI3_log_level { SAVAPI3_LOG_DEBUG = 0, SAVAPI3_LOG_INFO, SAVAPI3_LOG_WARNING, SAVAPI3_LOG_ALERT, SAVAPI3_LOG_ERROR }
  The enumeration used to specify the SAVAPI's logging levels.

5.20.4 Typedef Documentation

typedef struct SAVAPI3_alert_url_data SAVAPI3_ALERT_URL_DATA

Structure associated with the ALERTURL report.

typedef struct SAVAPI3_archive_open_data SAVAPI3_ARCHIVE_OPEN_DATA

Contains the data sent to a archive_open callback.

typedef struct SAVAPI3_callback_data SAVAPI3_CALLBACK_DATA

Structure passed by SAVAPI to a user defined callback, containing all the necessary data.

typedef struct SAVAPI3_command_data SAVAPI3_COMMAND_DATA

The structure to be passed when sending a command.

typedef struct SAVAPI3_error_data SAVAPI3_ERROR_DATA

The structure associated with report error callback.

The callback is triggered each time an error occurred on scanning process (an I/O error for instance). Also the callback can be called if warnings or infos reports during scanning are activated.

Note:

See SAVAPI3_CALLBACK_REPORT_ERROR

typedef struct SAVAPI3_file_info SAVAPI3_FILE_INFO

Contains data about the scanned file.

typedef struct SAVAPI3_file_status_data SAVAPI3_FILE_STATUS_DATA

Contains the data sent to a report file status callback.

Note:

See SAVAPI3_CALLBACK_REPORT_FILE_STATUS

typedef struct SAVAPI3_global_init SAVAPI3_GLOBAL_INIT

The structure used at SAVAPI initialization.

typedef struct SAVAPI3_iframe_url_data SAVAPI3_IFRAME_URL_DATA

Structure associated with the iframe report.

typedef struct SAVAPI3_instance_init SAVAPI3_INSTANCE_INIT

The structure used at SAVAPI instance creation.

typedef struct SAVAPI3_key_value SAVAPI3_KEY_VALUE

Generic container.

This kind of container is very useful in case of need to store many options. It offers a very elegant encapsulation and a very high flexibility (the user will not know how data will be stored).

The elements from container are accessed using a key (SAVAPI3_key_value::id member). The element is accessed through SAVAPI3_key_value::value member and its type through SAVAPI3_key_value::type member

typedef enum _SAVAPI3_log_level SAVAPI3_LOG_LEVEL

The enumeration used to specify the SAVAPI's logging levels.

typedef struct SAVAPI3_malware_info SAVAPI3_MALWARE_INFO

Contains data about the found malware in an infected/suspicious file.

typedef struct SAVAPI3_pre_scan_data SAVAPI3_PRESCAN_DATA

Contains the data sent to a prescan callback.

typedef struct SAVAPI3_repairable_data SAVAPI3_REPAIRABLE_DATA

Structure associated with the REPAIRABLE report.

typedef struct SAVAPI3_report_content_data SAVAPI3_REPORT_CONTENT_DATA

The structure associated with report content callback.

typedef struct SAVAPI3_report_progress_data SAVAPI3_REPORT_PROGRESS_DATA

The structure associated with report progress callback.

typedef struct SAVAPI3_report_scan_details_data SAVAPI3_REPORT_SCAN_DETAILS_DATA

The structure associated with report scan details callback.

typedef struct SAVAPI3_signal_data SAVAPI3_SIGNAL_DATA

The structure to be passed when sending a signal.

typedef struct SAVAPI3_version SAVAPI3_VERSION

The structure used to retrieve SAVAPI version.

5.20.5 Enumeration Type Documentation

enum _SAVAPI3_log_level

The enumeration used to specify the SAVAPI's logging levels.

Enumerator:

SAVAPI3_LOG_DEBUG Low level (debug, trace) messages. This the service MESSAGE level equivalent

SAVAPI3_LOG_INFO informative messages

SAVAPI3_LOG_WARNING warning messages

SAVAPI3_LOG_ALERT alert messages (i.e. malware found or any other alert)

SAVAPI3_LOG_ERROR error messages

5.21 SAVAPI typedefs

5.21.1 Typedefs

- typedef void * SAVAPI3_FD
  SAVAPI instance handle.
- typedef int(* SAVAPI3_CALLBACK)(SAVAPI3_CALLBACK_DATA *data)
  SAVAPI callback function pointer definition.
- typedef void(* SAVAPI3_LOG_CALLBACK)(SAVAPI3_LOG_LEVEL log_level, const SAVAPI_TCHAR *message, void *user_data)
  SAVAPI callback for logging.

5.21.2 Typedef Documentation

typedef int(* SAVAPI3_CALLBACK)(SAVAPI3_CALLBACK_DATA *data)

SAVAPI callback function pointer definition.

Parameters:

data [IN]: Pointer to the structure containing the callback data.

typedef void* SAVAPI3_FD

SAVAPI instance handle.

typedef void(* SAVAPI3_LOG_CALLBACK)(SAVAPI3_LOG_LEVEL log_level, const SAVAPI_TCHAR *message, void *user_data)

SAVAPI callback for logging.

Parameters:

log_level [IN]: The log level for the given message
message [IN]: The message to be logged
user_data [IN]: The user context

Returns:

Nothing

5.22 SAVAPI functions

5.22.1 Functions

- int SAVAPI3_EXP SAVAPI3_set_log_callback (SAVAPI3_LOG_CALLBACK log_fct, SAVAPI3_LOG_LEVEL min_level, void *user_data)
  Sets the SAVAPI logging function.
- int SAVAPI3_EXP SAVAPI3_initialize (SAVAPI3_GLOBAL_INIT *savapi_init)
  SAVAPI initialization function Initializes the SAVAPI library, according to the parameters specified in the initialization structure. It should be called once per process.
- int SAVAPI3_EXP SAVAPI3_uninitialize ()
  SAVAPI uninitialization function Uninitializes the SAVAPI library, cleaning up all used resources. Once called, all subsequent SAVAPI calls will fail with SAVAPI3_E_NOT_INITIALIZED error code.
- int SAVAPI3_EXP SAVAPI3_get_version (SAVAPI3_VERSION *version)
  Returns SAVAPI version.
- int SAVAPI3_EXP SAVAPI3_create_instance (SAVAPI3_INSTANCE_INIT *init, SAVAPI3_FD *savapi_fd)
  SAVAPI factory function The function opens a connection to the SAVAPI daemon for client-mode, or, for library mode, it creates a new SAVAPI instance.
- int SAVAPI3_EXP SAVAPI3_release_instance (SAVAPI3_FD *savapi_fd)
  Destroys a SAVAPI handler, previously created with SAVAPI3_create_instance . The function closes the connection to the SAVAPI daemon for client-mode, or, for library mode, it releases the SAVAPI instance.
- int SAVAPI3_EXP SAVAPI3_set_user_data (SAVAPI3_FD *savapi_fd, void *user_data)
  Sets user specific data. This functions sets user data that will be returned untouched as user_data member of SAVAPI3_CALLBACK_DATA structure.
- int SAVAPI3_EXP SAVAPI3_is_running ()
  Determines if the SAVAPI daemon is running The library must be initialized with the proper daemon connection parameters for this function to run correctly.
- int SAVAPI3_EXP SAVAPI3_is_running_ex (const SAVAPI_TCHAR *hostname, unsigned int port)
  Determines if the SAVAPI daemon is running on the specified interface (hostname and port).
- int SAVAPI3_EXP SAVAPI3_register_callback (SAVAPI3_FD *savapi_fd, unsigned int callback_id, SAVAPI3_CALLBACK callback)
  Registers a client defined callback.
- int SAVAPI3_EXP SAVAPI3_unregister_callback (SAVAPI3_FD *savapi_fd, unsigned int callback_id, SAVAPI3_CALLBACK callback)
  Unregisters a previously registered client defined callback.
- int SAVAPI3_EXP SAVAPI3_scan (SAVAPI3_FD *savapi_fd, SAVAPI_TCHAR *file_name)
  Starts a scanning process. During the scan operation the registered callbacks may be triggered.
- int SAVAPI3_EXP SAVAPI3_set (SAVAPI3_FD *savapi_fd, unsigned int option_id, SAVAPI_TCHAR *buffer)
  Sets SAVAPI individual settings.

- int SAVAPI3_EXP SAVAPI3_get (SAVAPI3_FD *savapi_fd, unsigned int option_id, SAVAPI_TCHAR *buffer, SAVAPI_SIZE_T *buffer_size)
  Reads SAVAPI settings.
- int SAVAPI3_EXP SAVAPI3_get_dynamic_detect (SAVAPI_TCHAR *type, int *id)
  Retrieve the various types that can be detected (and dynamically turned on/off).
- int SAVAPI3_EXP SAVAPI3_send_signal (SAVAPI3_FD *savapi_fd, unsigned int signal_id, SAVAPI3_SIGNAL_DATA *data)
  Sends a signal to a specific SAVAPI instance The SAVAPI3_scan may take a long amount of time to finish scanning its target and in some situations a forced abort would be desirable. In these kind of situations, SAVAPI3_send_signal may help by sending signals to a running SAVAPI instance (SAVAPI3_SIGNAL_SCAN_ABORT for instance).
- int SAVAPI3_EXP SAVAPI3_set_fops (SAVAPI3_FD *savapi_fd, void *fops_pointer, void *fops_context)
  Specify the new fops who will be used by the engine for reading.
- void SAVAPI3_EXP SAVAPI3_free (void **ptr)
  Frees the memory space pointed to by ptr.
- int SAVAPI3_EXP SAVAPI3_reload_engine ()
  Reloads the engine from the location given at global initialization.
- int SAVAPI3_EXP SAVAPI3_reload_engine_ex (const SAVAPI3_GLOBAL_INIT *global_init)
  Reloads the engine from the specified location.

5.22.2 Function Documentation

int SAVAPI3_EXP SAVAPI3_create_instance (SAVAPI3_INSTANCE_INIT * *init*, SAVAPI3_FD * *savapi_fd*)

SAVAPI factory function The function opens a connection to the SAVAPI daemon for client-mode, or, for library mode, it creates a new SAVAPI instance.

Parameters:

init [IN]: Pointer to a structure containing all the initialization data needed to create a SAVAPI instance.
savapi_fd [OUT]: Handle to the SAVAPI instance. To be used in all the subsequent SAVAPI calls

Returns:

Null if everything went OK or an error code otherwise

void SAVAPI3_EXP SAVAPI3_free (void ** *ptr*)

Frees the memory space pointed to by ptr.

Parameters:

ptr [IN/OUT]: Pointer who will become free and null

Returns:

Nothing

int SAVAPI3_EXP SAVAPI3_get (SAVAPI3_FD * *savapi_fd*, unsigned int *option_id*, SAVAPI_TCHAR * *buffer*, SAVAPI_SIZE_T * *buffer_size*)

Reads SAVAPI settings.

Parameters:

savapi_fd [IN]: Pointer to a SAVAPI instance
option_id [IN]: The id of the option to be retrieved
buffer [OUT]: Buffer allocated by caller which will store the result of a successful get as a NULL terminated string. If the buffer is NULL and the other parameters are valid, the function will set the needed buffer-size and return SAVAPI3_S_OK
buffer_size [IN/OUT]: Specifies the size, given in SAVAPI_TCHAR characters, of the buffer argument. If the buffer is not large enough, upon return it will contain the needed size. If it's equal or larger than the needed size, it will remain unchanged.

Returns:

SAVAPI3_S_OK If everything went ok an error code otherwise.

int SAVAPI3_EXP SAVAPI3_get_dynamic_detect (SAVAPI_TCHAR * *type*, int * *id*)

Retrieve the various types that can be detected (and dynamically turned on/off).

Parameters:

type [IN]: The type that should be detected (current values: ADSPY,BDC,DIAL,GAME,HEUR_DBLEXT,JOKE,PCK,SPR)
id [OUT]: Stores the type id

Returns:

SAVAPI3_S_OK If everything went ok an error code otherwise.

Warning:

Not implemented yet.

int SAVAPI3_EXP SAVAPI3_get_version (SAVAPI3_VERSION * *version*)

Returns SAVAPI version.

Parameters:

version [OUT]: Pointer to the structure where to store the result

Returns:

SAVAPI3_S_OK on success and an error otherwise

Note:

If this function is called before SAVAPI3_initialize() , the SAVAPI3_E_NOT_INITIALIZED will be returned instead.

int SAVAPI3_EXP SAVAPI3_initialize (SAVAPI3_GLOBAL_INIT * *savapi_init*)

SAVAPI initialization function Initializes the SAVAPI library, according to the parameters specified in the initialization structure. It should be called once per process.

Parameters:

savapi_init [IN]: A pointer to the initialization structure, which must be filled with the proper values for initialization

Returns:

Null if success or an error code otherwise

Note:

The initialization function must be called a single time per process and before calling any other SAVAPI function (except the SAVAPI3_set_log_callback() function).

int SAVAPI3_EXP SAVAPI3_is_running ()

Determines if the SAVAPI daemon is running The library must be initialized with the proper daemon connection parameters for this function to run correctly.

Returns:

- 0 If the daemon is stopped
- 1 If the daemon is running
- SAVAPI3_E_NOT_INITIALIZED If the SAVAPI library was not properly initialized.

Note:

This function is deprecated, use SAVAPI3_is_running_ex() instead.

int SAVAPI3_EXP SAVAPI3_is_running_ex (const SAVAPI_TCHAR * *hostname*, unsigned int *port*)

Determines if the SAVAPI daemon is running on the specified interface (hostname and port).

Parameters:

hostname [IN]: Specifies the host on which the SAVAPI daemon is located.
port [IN]: Specifies the port on which to connect to the daemon.

Returns:

SAVAPI3_S_OK if the SAVAPI daemon is running on the given interface or an error code otherwise.

Note:

For local sockets (see SAVAPI3_FLAG_USE_LOCAL_SOCKET) the

Parameters:

hostname must be a path to a file and the
port must be 0.

Note:

This function returns SAVAPI3_E_NOT_SUPPORTED in library mode.

int SAVAPI3_EXP SAVAPI3_register_callback (SAVAPI3_FD * *savapi_fd*, unsigned int *callback_id*, SAVAPI3_CALLBACK *callback*)

Registers a client defined callback.

Parameters:

savapi_fd [IN]: Handle to the SAVAPI instance on which the callback will be available.
callback_id [IN]: The callback type (e.g. SAVAPI3_CALLBACK_REPORT_FILE_STATUS, SAVAPI3_CALLBACK_ARCHIVE_OPEN etc.)
callback [IN]: Pointer to a callback function

Returns:

SAVAPI3_S_OK if everything went OK or an error code otherwise

Note:

Callback registering is not allowed during scanning operations, otherwise the SAVAPI3_E_BUSY will be returned.
Only one callback function is allowed to be registered per callback type/id. If for the given type/id a callback function was already registered then this function will return SAVAPI3_E_INVALID_PARAMETER

See also:

Callbacks' ids

int SAVAPI3_EXP SAVAPI3_release_instance (SAVAPI3_FD * *savapi_fd*)

Destroys a SAVAPI handler, previously created with SAVAPI3_create_instance . The function closes the connection to the SAVAPI daemon for client-mode, or, for library mode, it releases the SAVAPI instance.

Parameters:

savapi_fd [IN/OUT]: SAVAPI instance to be released. As a precaution, the pointer will be nulled in order to become very clear that pointer will be unusable for now on

Returns:

Null if everything went OK or an error code otherwise

Note:

- For each handler created with SAVAPI3_create_instance function, the correspondent **`SAVAPI3_release_instance`** `must` be called!
- After calling the function the **`savapi_fd`** `pointer` will be invalid and must not be used anymore

int SAVAPI3_EXP SAVAPI3_reload_engine ()

Reloads the engine from the location given at global initialization.

Returns:

SAVAPI3_S_OK or an error code

Note:

This function will simply call the SAVAPI3_uninitialize routine followed by the SAVAPI3_initialize having as parameter the same data provided at global initialization (aka the call to SAVAPI3_initialize) in order to (re)load the engine files from the initial location(s).
All constraints that apply to the SAVAPI3_uninitialize and SAVAPI3_initialize are also available for this function. Therefore, in order to call the function, all instances must be released otherwise the SAVAPI3_E_BUSY will be returned.
For better functionality and the possibility to load a new engine without interrupt of service, please check the SAVAPI3_reload_engine_ex function.
This function will return SAVAPI3_E_NOT_SUPPORTED in client-mode

Warning:

If something wrong goes when the new engine is loaded (e.g. engine is corrupted, some engine files are missing, etc), the library will not be usable anymore (i.e. all functions will return

SAVAPI3_E_NOT_INITIALIZED) so the user should only call SAVAPI3_uninitialize and abort the execution.

int SAVAPI3_EXP SAVAPI3_reload_engine_ex (const SAVAPI3_GLOBAL_INIT * *global_init*)

Reloads the engine from the specified location.

Parameters:

global_init [IN]: A pointer to the initialization structure containing the paths to the new engine and vdf files

Returns:

SAVAPI3_S_OK or an error code

Note:

When this function is called, the engine will be (re)loaded from the specified path. The old engine's instances will be kept until their reference counter will reach 0. Calling this function, affects only the new SAVAPI instances (obtained by calling SAVAPI3_create_instance()). They will use the new loaded engine. The SAVAPI instances that are already started won't be affected by this function, they will continue to use the engine that they were started with.
This function will return SAVAPI3_E_NOT_SUPPORTED in client-mode

int SAVAPI3_EXP SAVAPI3_scan (SAVAPI3_FD * *savapi_fd*, SAVAPI_TCHAR * *file_name*)

Starts a scanning process. During the scan operation the registered callbacks may be triggered.

Parameters:

savapi_fd [IN]: The handle of the SAVAPI instance that will do the scanning
file_name [IN]: The name of the file to be scanned.

Returns:

Null in case of success or an error code otherwise

Note:

The execution will not leave SAVAPI3_scan function until scan process is finished.
SAVAPI supports various scan types depending on the file_name format:

- 'path/to/the/file/on/disk' for normal file scanning.
- 'mem://0xAddress,size,name' for scan in memory. The '0xAddress' is the memory area where the buffer with size 'size' is loaded. The 'name' is the display name used when callbacks are triggered.
- 'hex_enc://hex_encoded_filename' for scanning files with filename given using hex encoding. This is useful for special encodings (ex. Chinese) in order to avoid conversions.

int SAVAPI3_EXP SAVAPI3_send_signal (SAVAPI3_FD * *savapi_fd*, unsigned int *signal_id*, SAVAPI3_SIGNAL_DATA * *data*)

Sends a signal to a specific SAVAPI instance The SAVAPI3_scan may take a long amount of time to finish scanning its target and in some situations a forced abort would be desirable. In these kind of situations, SAVAPI3_send_signal may help by sending signals to a running SAVAPI instance (SAVAPI3_SIGNAL_SCAN_ABORT for instance).

Parameters:

savapi_fd [IN]: SAVAPI instance
signal_id [IN]: Identifies the signal to be sent. See Signal IDs section
data [IN]: Specific data to be sent when sending the signal. See SAVAPI3_SIGNAL_DATA

Returns:

SAVAPI3_S_OK for success or an error code otherwise.

Note:

The SAVAPI signals were designed to be sent asynchronously, when an event arrives (Ctrl+C was issued for instance) and if program execution is within SAVAPI3_scan . It makes no sense to send a signal to a SAVAPI instance when we have execution flow control.

See also:

Current supported SAVAPI signals

int SAVAPI3_EXP SAVAPI3_set (SAVAPI3_FD * *savapi_fd*, unsigned int *option_id*, SAVAPI_TCHAR * *buffer*)

Sets SAVAPI individual settings.

Parameters:

savapi_fd [IN]: Pointer to a SAVAPI instance
option_id [IN]: The id of the option to be set
buffer [IN]: NULL-terminated string containing the value of the option to be set

Returns:

SAVAPI3_S_OK If everything went ok an error code otherwise.

Note:

Calling this function during a scanning operation performed by this instance will fail.

int SAVAPI3_EXP SAVAPI3_set_fops (SAVAPI3_FD * *savapi_fd*, void * *fops_pointer*, void * *fops_context*)

Specify the new fops who will be used by the engine for reading.

Parameters:

savapi_fd [IN]: SAVAPI instance
fops_pointer [IN]: Pointer to the fops to use in the current savapi session
fops_context [IN]: This context will be passed back to the application by each call to the fops used

Returns:

SAVAPI3_S_OK for success or an error code otherwise.

Note:

This function will return SAVAPI3_E_NOT_SUPPORTED in client-mode

int SAVAPI3_EXP SAVAPI3_set_log_callback (SAVAPI3_LOG_CALLBACK *log_fct*, SAVAPI3_LOG_LEVEL *min_level*, void * *user_data*)

Sets the SAVAPI logging function.

Parameters:

log_fct [IN]: The function used for logging. If given function is NULL all data set with a previous SAVAPI3_set_log_callback call will be cleared so logging will not be performed anymore.
min_level [IN]: Sets the desired minimum log level. This can be used to filter unwanted log-levels, so that if a message have a lower level, it will be automatically "thrown" by the SAVAPI
user_data [IN]: The user context

Returns:

Null if success or an error code otherwise

Note:

This function can be called before or/and after global initialization (calling before it's recommended, so that any error messages in the SAVAPI3_initialize() can be logged)
This can be called several times in the same process, so that SAVAPI's user can change the log-level on-the-fly

int SAVAPI3_EXP SAVAPI3_set_user_data (SAVAPI3_FD * *savapi_fd*, void * *user_data*)

Sets user specific data. This functions sets user data that will be returned untouched as *user_data* member of SAVAPI3_CALLBACK_DATA structure.

Parameters:

savapi_fd [IN/OUT]: Handle to the SAVAPI instance.
user_data [IN]: user specific data

Returns:

SAVAPI3_S_OK in case of success or an error code otherwise

Note:

The user is responsible with the memory management. This function will only set the value given in the SAVAPI3_CALLBACK_DATA structure. It will not reserve or free memory for the data.

int SAVAPI3_EXP SAVAPI3_uninitialize ()

SAVAPI uninitialization function Uninitializes the SAVAPI library, cleaning up all used resources. Once called, all subsequent SAVAPI calls will fail with SAVAPI3_E_NOT_INITIALIZED error code.

Returns:

Null if everything went OK or an error code otherwise

Note:

The uninitialization function must be called a single time per process and must be last called SAVAPI function.
All SAVAPI instances must be released before calling this function, otherwise the SAVAPI3_E_BUSY will be returned.

int SAVAPI3_EXP SAVAPI3_unregister_callback (SAVAPI3_FD * *savapi_fd*, unsigned int *callback_id*, SAVAPI3_CALLBACK *callback*)

Unregisters a previously registered client defined callback.

Parameters:

savapi_fd [IN]: Handle to the SAVAPI instance.
callback_id [IN]: The callback type (e.g. SAVAPI3_CALLBACK_REPORT_FILE_STATUS, SAVAPI3_CALLBACK_ARCHIVE_OPEN etc.)
callback [IN]: Pointer to a callback function

Returns:

SAVAPI3_S_OK in case of success or an error code otherwise

Note:

Callback unregistering is not allowed during scanning operations, otherwise the SAVAPI3_E_BUSY will be returned.
The callback type/id is searched by SAVAPI in its internal callback list and if found then the callback function will be removed, otherwise this function will return SAVAPI3_E_INVALID_PARAMETER

6 Data Structure Documentation

6.1 SAVAPI3_report_content_data::_content_data Union Reference

6.1.1 Data Fields

- SAVAPI3_IFRAME_URL_DATA * iframeurl_data

6.1.2 Detailed Description

Union used to switch the content data depending on the received 'type'

6.1.3 Field Documentation

SAVAPI3_IFRAME_URL_DATA* SAVAPI3_report_content_data::_content_data::iframeurl_data

Information about the iframe report (url, attribute)

The documentation for this union was generated from the following file:

- savapi3.h

6.2 SAVAPI3_report_scan_details_data::_scan_details_data Union Reference

6.2.1 Data Fields

- SAVAPI3_ALERT_URL_DATA * alert_url_data
- SAVAPI3_REPAIRABLE_DATA * repairable_data

6.2.2 Detailed Description

Union used to switch the scan details data depending on the received 'type'

6.2.3 Field Documentation

SAVAPI3_ALERT_URL_DATA*
SAVAPI3_report_scan_details_data::_scan_details_data::alert_url_data

Contains the alert URL report

SAVAPI3_REPAIRABLE_DATA* SAVAPI3_report_scan_details_data::_scan_details_data::repairable_data

Contains the details about the reparable data

The documentation for this union was generated from the following file:

- savapi3.h

6.3 SAVAPI3_alert_url_data Struct Reference

Structure associated with the ALERTURL report.

6.3.1 Data Fields

- SAVAPI_TCHAR * alert_url
- SAVAPI3_FILE_INFO file_info

6.3.2 Detailed Description

Structure associated with the ALERTURL report.

6.3.3 Field Documentation

SAVAPI_TCHAR* SAVAPI3_alert_url_data::alert_url

Pointer to the string containing the alert URL

SAVAPI3_FILE_INFO SAVAPI3_alert_url_data::file_info

Information (name, type, level) about the scanned file

The documentation for this struct was generated from the following file:

- savapi3.h

6.4 SAVAPI3_archive_open_data Struct Reference

Contains the data sent to a archive_open callback.

6.4.1 Data Fields

- unsigned int flags
- SAVAPI3_FILE_INFO file_info

6.4.2 Detailed Description

Contains the data sent to a archive_open callback.

6.4.3 Field Documentation

SAVAPI3_FILE_INFO SAVAPI3_archive_open_data::file_info

Information (name, type, level) about the archive to be opened

unsigned int SAVAPI3_archive_open_data::flags

General purpose flags field.

Note:

Currently defined flags: none.

The documentation for this struct was generated from the following file:

- savapi3.h

6.5 SAVAPI3_callback_data Struct Reference

Structure passed by SAVAPI to a user defined callback, containing all the necessary data.

6.5.1 Data Structures

- union specific_data

6.5.2 *Callbacks specific data.* Data Fields

- unsigned int type
- unsigned int version
- unsigned int flags
- void * user_data

  User custom data.
- union SAVAPI3_callback_data::specific_data callback_data

  Callbacks specific data.

6.5.3 Detailed Description

Structure passed by SAVAPI to a user defined callback, containing all the necessary data.

6.5.4 Field Documentation

union SAVAPI3_callback_data::specific_data SAVAPI3_callback_data::callback_data

Callbacks specific data.

unsigned int SAVAPI3_callback_data::flags

Reserved

unsigned int SAVAPI3_callback_data::type

The callback id. See Callbacks' ids

void* SAVAPI3_callback_data::user_data

User custom data.

Note:

SAVAPI will not make any assumption regarding this field. It will just be passed back to callback function

unsigned int SAVAPI3_callback_data::version

The callback version

The documentation for this struct was generated from the following file:

- savapi3.h

6.6 SAVAPI3_command_data Struct Reference

The structure to be passed when sending a command.

6.6.1 Data Fields

- unsigned int signal_id
- void * command_data
  Signal specific data.

6.6.2 Detailed Description

The structure to be passed when sending a command.

6.6.3 Field Documentation

void* SAVAPI3_command_data::command_data

Signal specific data.

Note:

Currently SAVAPI has defined only SAVAPI3_SIGNAL_SCAN_ABORT signal which doesn't require any data. Thus, "specific_data" field is currently empty.

Todo:

Add specific date as soon as new signals, which require data will be defined.

unsigned int SAVAPI3_command_data::signal_id

signal id. See SAVAPI signals

The documentation for this struct was generated from the following file:

- savapi3.h

6.7 SAVAPI3_error_data Struct Reference

The structure associated with report error callback.

6.7.1 Data Fields

- SAVAPI3_FILE_INFO file_info
- unsigned int category
- unsigned int level
- int code
- SAVAPI3_KEY_VALUE * options

6.7.2 Detailed Description

The structure associated with report error callback.

The callback is triggered each time an error occurred on scanning process (an I/O error for instance). Also the callback can be called if warnings or infos reports during scanning are activated.

Note:

See SAVAPI3_CALLBACK_REPORT_ERROR

6.7.3 Field Documentation

unsigned int SAVAPI3_error_data::category

error category see Error categories

int SAVAPI3_error_data::code

error code. See SAVAPI return codes

Note:

If error level is not SAVAPI3_ELEVEL_ERROR this field contains flags. See Scan warnings and Scan informations

SAVAPI3_FILE_INFO SAVAPI3_error_data::file_info

Information (name, type, level) about the file where the error occured

unsigned int SAVAPI3_error_data::level

error level see Error levels

SAVAPI3_KEY_VALUE* SAVAPI3_error_data::options

The container contain currently only the error code string.

Note:

The code is the id within container

The documentation for this struct was generated from the following file:

- savapi3.h

6.8 SAVAPI3_file_info Struct Reference

Contains data about the scanned file.

6.8.1 Data Fields

- SAVAPI_TCHAR * name
- unsigned int type
- unsigned int level

6.8.2 Detailed Description

Contains data about the scanned file.

6.8.3 Field Documentation

unsigned int SAVAPI3_file_info::level

The file recursion level (0 for regular files, +1 for each level in an archive)

SAVAPI_TCHAR* SAVAPI3_file_info::name

file name

unsigned int SAVAPI3_file_info::type

The file type. See File types . Note: can be one or more types (ex: file is an archive and is found in an archive).

The documentation for this struct was generated from the following file:

- savapi3.h

6.9 SAVAPI3_file_status_data Struct Reference

Contains the data sent to a report file status callback.

6.9.1 Data Fields

- unsigned int flags
- unsigned int scan_answer
- SAVAPI3_FILE_INFO file_info
- SAVAPI3_MALWARE_INFO malware_info
  Malware information (name, type, etc). See SAVAPI3_malware_info for more details.
- unsigned int warning
- unsigned int info

6.9.2 Detailed Description

Contains the data sent to a report file status callback.

Note:
See SAVAPI3_CALLBACK_REPORT_FILE_STATUS

6.9.3 Field Documentation

SAVAPI3_FILE_INFO SAVAPI3_file_status_data::file_info

File information (name, type, level). See SAVAPI3_file_info for more details

unsigned int SAVAPI3_file_status_data::flags

General purpose flags field.

Note:
Currently defined flags: none.

unsigned int SAVAPI3_file_status_data::info

additional info to report See Scan informations

SAVAPI3_MALWARE_INFO SAVAPI3_file_status_data::malware_info

Malware information (name, type, etc). See SAVAPI3_malware_info for more details.

Note:
Contains data only if the object processed is not clean.

unsigned int SAVAPI3_file_status_data::scan_answer

File scan answer. See SAVAPI scan statuses for available values

unsigned int SAVAPI3_file_status_data::warning

warning value to report See Scan warnings

The documentation for this struct was generated from the following file:

- savapi3.h

6.10 SAVAPI3_global_init Struct Reference

The structure used at SAVAPI initialization.

6.10.1 Data Fields

- unsigned int program_type
- SAVAPI_TCHAR * engine_dirpath
- SAVAPI_TCHAR * vdfs_dirpath
- SAVAPI_TCHAR * avll_dirpath
- SAVAPI_TCHAR * key_file_name
- SAVAPI_TCHAR * key_dir

6.10.2 Detailed Description

The structure used at SAVAPI initialization.

6.10.3 Field Documentation

SAVAPI_TCHAR* SAVAPI3_global_init::avll_dirpath

IGNORED OPTION - Path to a directory containing avll license library

SAVAPI_TCHAR* SAVAPI3_global_init::engine_dirpath

Path to a directory containing engine modules

SAVAPI_TCHAR* SAVAPI3_global_init::key_dir

Full path to a folder containing one or more license file(s) with extension .key

SAVAPI_TCHAR* SAVAPI3_global_init::key_file_name

IGNORED OPTION - Full path to a license file.

unsigned int SAVAPI3_global_init::program_type

The unique program number which identifies the 3rd party application for the license checking function. This has to be requested from Avira.

SAVAPI_TCHAR* SAVAPI3_global_init::vdfs_dirpath

Path to a directory containing the signature files

The documentation for this struct was generated from the following file:

- savapi3.h

6.11 SAVAPI3_iframe_url_data Struct Reference

Structure associated with the iframe report.

6.11.1 Data Fields

- unsigned int attribute
- SAVAPI_TCHAR * url

6.11.2 Detailed Description

Structure associated with the iframe report.

6.11.3 Field Documentation

unsigned int SAVAPI3_iframe_url_data::attribute

iframe attribute. See Iframes informations

SAVAPI_TCHAR* SAVAPI3_iframe_url_data::url

iframe url.

The documentation for this struct was generated from the following file:

- savapi3.h

6.12 SAVAPI3_instance_init Struct Reference

The structure used at SAVAPI instance creation.

6.12.1 Data Fields

- unsigned int flags
- unsigned int connection_timeout
- SAVAPI_TCHAR * host_name
- unsigned int port
- unsigned int scan_timeout
- unsigned int get_timeout
- unsigned int set_timeout
- SAVAPI_TCHAR * username
- SAVAPI_TCHAR * password

6.12.2 Detailed Description

The structure used at SAVAPI instance creation.

6.12.3 Field Documentation

unsigned int SAVAPI3_instance_init::connection_timeout

Specified the connection timeout in milliseconds.

Note:
Available only in client-mode.

unsigned int SAVAPI3_instance_init::flags

Initialization flags, right now only the flag deciding the connection type is defined. See Initialization flags

unsigned int SAVAPI3_instance_init::get_timeout

Specifies the timeout (in milliseconds) of the get options operation.

Note:
Used only in client-mode.

SAVAPI_TCHAR* SAVAPI3_instance_init::host_name

Specifies the machine on which the SAVAPI daemon is located.

Note:
Used only in client-mode.

SAVAPI_TCHAR* SAVAPI3_instance_init::password

This field is not used.

Note:
Any data contained by this field will be ignored.

Warning:
Please note that this field will be removed in a future release.

unsigned int SAVAPI3_instance_init::port

Specifies the port on which to connect to the daemon.

Note:
Used in the same conditions as host_name .

unsigned int SAVAPI3_instance_init::scan_timeout

Specifies the timeout (in milliseconds) of the scan operation.

Note:
Used only in client-mode.

unsigned int SAVAPI3_instance_init::set_timeout

Specifies the timeout (in milliseconds) of the set options operation.

Note:
Used only in client-mode.

SAVAPI_TCHAR* SAVAPI3_instance_init::username

This field is not used.

Note:
Any data contained by this field will be ignored.

Warning:
Please note that this field will be removed in a future release.

The documentation for this struct was generated from the following file:

- savapi3.h

6.13 SAVAPI3_key_value Struct Reference

Generic container.

6.13.1 Data Fields

- unsigned int id
- unsigned int type
- char * value

6.13.2 Detailed Description

Generic container.

This kind of container is very useful in case of need to store many options. It offers a very elegant encapsulation and a very high flexibility (the user will not know how data will be stored).

The elements from container are accessed using a key (SAVAPI3_key_value::id member). The element is accessed through SAVAPI3_key_value::value member and its type through SAVAPI3_key_value::type member

---

6.13.3 Field Documentation

unsigned int SAVAPI3_key_value::id

The element associated id

unsigned int SAVAPI3_key_value::type

The element type

char* SAVAPI3_key_value::value

The element value

The documentation for this struct was generated from the following file:

- savapi3.h

6.14 SAVAPI3_malware_info Struct Reference

Contains data about the found malware in an infected/suspicious file.

6.14.1 Data Fields

- SAVAPI_TCHAR * name
- SAVAPI_TCHAR * type
- SAVAPI_TCHAR * message
- SAVAPI_TCHAR * app_flags
- unsigned int removable
- unsigned short strict

6.14.2 Detailed Description

Contains data about the found malware in an infected/suspicious file.

6.14.3 Field Documentation

SAVAPI_TCHAR* SAVAPI3_malware_info::app_flags

mallware flags

SAVAPI_TCHAR* SAVAPI3_malware_info::message

Additional information about found malware

SAVAPI_TCHAR* SAVAPI3_malware_info::name

The malware name or null if file is clean

unsigned int SAVAPI3_malware_info::removable

1 if malware removable/ 0 otherwise

unsigned short SAVAPI3_malware_info::strict

Malware signature found at correct offset

SAVAPI_TCHAR* SAVAPI3_malware_info::type

Kind of the found malware

The documentation for this struct was generated from the following file:

- savapi3.h

6.15 SAVAPI3_pre_scan_data Struct Reference

Contains the data sent to a prescan callback.

6.15.1 Data Fields

- unsigned int flags
- SAVAPI3_FILE_INFO file_info

6.15.2 Detailed Description

Contains the data sent to a prescan callback.

6.15.3 Field Documentation

SAVAPI3_FILE_INFO SAVAPI3_pre_scan_data::file_info

Information (name, type, level) about the scanned file

unsigned int SAVAPI3_pre_scan_data::flags

General purpose flags field.

Note:

Currently defined flags: none.

The documentation for this struct was generated from the following file:

- savapi3.h

6.16 SAVAPI3_repairable_data Struct Reference

Structure associated with the REPAIRABLE report.

6.16.1 Data Fields

- SAVAPI3_FILE_INFO file_info
- SAVAPI3_MALWARE_INFO malware_info

6.16.2 Detailed Description

Structure associated with the REPAIRABLE report.

6.16.3 Field Documentation

SAVAPI3_FILE_INFO SAVAPI3_repairable_data::file_info

Information (name, type, level) about the scanned file

SAVAPI3_MALWARE_INFO SAVAPI3_repairable_data::malware_info

Malware information (name, type, etc).

The documentation for this struct was generated from the following file:

- savapi3.h

6.17 SAVAPI3_report_content_data Struct Reference

The structure associated with report content callback.

6.17.1 Data Structures

- union _content_data

6.17.2 Data Fields

- unsigned int flags
- unsigned int type
- SAVAPI3_FILE_INFO file_info
- union SAVAPI3_report_content_data::_content_data content_data

---

6.17.3 Detailed Description

The structure associated with report content callback.

6.17.4 Field Documentation

union SAVAPI3_report_content_data::_content_data SAVAPI3_report_content_data::content_data

Union used to switch the content data depending on the received 'type'

SAVAPI3_FILE_INFO SAVAPI3_report_content_data::file_info

Information (name, type, level) about the scanned file

unsigned int SAVAPI3_report_content_data::flags

Reserved

unsigned int SAVAPI3_report_content_data::type

report type (SAVAPI report content types)

The documentation for this struct was generated from the following file:

- savapi3.h

6.18 SAVAPI3_report_progress_data Struct Reference

The structure associated with report progress callback.

6.18.1 Data Fields

- unsigned int flags
- SAVAPI_TCHAR * message

6.18.2 Detailed Description

The structure associated with report progress callback.

6.18.3 Field Documentation

unsigned int SAVAPI3_report_progress_data::flags

Reserved

SAVAPI_TCHAR* SAVAPI3_report_progress_data::message

the progress message

The documentation for this struct was generated from the following file:

- savapi3.h

6.19 SAVAPI3_report_scan_details_data Struct Reference

The structure associated with report scan details callback.

6.19.1 Data Structures

- union _scan_details_data

6.19.2 Data Fields

- unsigned int flags
- unsigned int type
- union SAVAPI3_report_scan_details_data::_scan_details_data scan_details_data

6.19.3 Detailed Description

The structure associated with report scan details callback.

6.19.4 Field Documentation

unsigned int SAVAPI3_report_scan_details_data::flags

Reserved

union SAVAPI3_report_scan_details_data::_scan_details_data SAVAPI3_report_scan_details_data::scan_details_data

Union used to switch the scan details data depending on the received 'type'

unsigned int SAVAPI3_report_scan_details_data::type

report type(see SAVAPI report scan details types)

The documentation for this struct was generated from the following file:

- savapi3.h

6.20 SAVAPI3_signal_data Struct Reference

The structure to be passed when sending a signal.

6.20.1 Data Fields

- unsigned int signal_id
- void * signal_data
  Signal specific data.

6.20.2 Detailed Description

The structure to be passed when sending a signal.

6.20.3 Field Documentation

void* SAVAPI3_signal_data::signal_data

Signal specific data.

Note:

Currently SAVAPI has defined only SAVAPI3_SIGNAL_SCAN_ABORT signal which doesn't require any data. Thus, "specific_data" field is currently empty.

Todo:

Add specific date as soon as new signals, which require data will be defined.

unsigned int SAVAPI3_signal_data::signal_id

signal id. See SAVAPI signals

The documentation for this struct was generated from the following file:

- savapi3.h

6.21 SAVAPI3_version Struct Reference

The structure used to retrieve SAVAPI version.

6.21.1 Data Fields

- unsigned int major
- unsigned int minor
- unsigned int build_major
- unsigned int build_minor

6.21.2 Detailed Description

The structure used to retrieve SAVAPI version.

6.21.3 Field Documentation

unsigned int SAVAPI3_version::build_major

Major version of the build

unsigned int SAVAPI3_version::build_minor

Minor version of the build

unsigned int SAVAPI3_version::major

Major version of the product

unsigned int SAVAPI3_version::minor

Minor version of the product

The documentation for this struct was generated from the following file:

- savapi3.h

6.22 SAVAPI3_callback_data::specific_data Union Reference

Callbacks specific data.

6.22.1 Data Fields

- SAVAPI3_PRESCAN_DATA * pre_scan_data
- SAVAPI3_ARCHIVE_OPEN_DATA * archive_open_data
- SAVAPI3_FILE_STATUS_DATA * file_status_data
- SAVAPI3_ERROR_DATA * error_data
- SAVAPI3_REPORT_PROGRESS_DATA * report_progress_data
- SAVAPI3_REPORT_CONTENT_DATA * report_content_data
- SAVAPI3_REPORT_SCAN_DETAILS_DATA * report_scan_details_data
- void * private_data

6.22.2 Detailed Description

Callbacks specific data.

6.22.3 Field Documentation

SAVAPI3_ARCHIVE_OPEN_DATA* SAVAPI3_callback_data::specific_data::archive_open_data

specific data for archive open callback

See SAVAPI3_archive_open_data

SAVAPI3_ERROR_DATA* SAVAPI3_callback_data::specific_data::error_data

specific data for error report callback

See SAVAPI3_error_data

SAVAPI3_FILE_STATUS_DATA* SAVAPI3_callback_data::specific_data::file_status_data

specific data for file status callback

See SAVAPI3_file_status_data

SAVAPI3_PRESCAN_DATA* SAVAPI3_callback_data::specific_data::pre_scan_data

specific data for pre scan callback

See SAVAPI3_pre_scan_data

void* SAVAPI3_callback_data::specific_data::private_data

private data. Reserved for internal use

**SAVAPI3_REPORT_CONTENT_DATA*
SAVAPI3_callback_data::specific_data::report_content_data**

specific data for report content callback. See SAVAPI3_report_content_data

**SAVAPI3_REPORT_PROGRESS_DATA*
SAVAPI3_callback_data::specific_data::report_progress_data**

specific data for report progress callback

See SAVAPI3_report_progress_data

SAVAPI3_REPORT_SCAN_DETAILS_DATA* SAVAPI3_callback_data::specific_data::report_scan_details_data

specific data for report scan details callback. See SAVAPI3_report_scan_details_data

---

The documentation for this union was generated from the following file:

- savapi3.h

7 File Documentation

7.1 asc_bin.h File Reference

7.1.1 Functions

- SAVAPI3_EXP int bin2asc (const char *binblock, char **ascblock)
  Convert the file name to ASCII hex representation.
- SAVAPI3_EXP int asc2bin (const char *ascblock, SAVAPI_SIZE_T ascblock_size, char **binblock)
  Convert ASCII hex representation to file name.
- SAVAPI3_EXP int bin2hex (const char *binblock, SAVAPI_SIZE_T binblock_size, char *hexblock, SAVAPI_SIZE_T *hexblock_size)
  Transform the string given in binblock in the equivalent encoded string in hexblock.
- SAVAPI3_EXP int hex2bin (const char *hexblock, SAVAPI_SIZE_T hexblock_size, char *binblock, SAVAPI_SIZE_T *binblock_size)
  Converts a series of hex-encoded characters to normal binary encoding.

7.1.2 Function Documentation

SAVAPI3_EXP int asc2bin (const char * *ascblock*, SAVAPI_SIZE_T *ascblock_size*, char ** *binblock*)

Convert ASCII hex representation to file name.

Parameters:

ascblock [IN]: The ASCII hex block
ascblock_size [IN]: The ASCII hex block size
binblock [OUT]: The converted file name

Returns:

SAVAPI3_S_OK for success or an error code

Note:

The binblock parameter will be internally allocated. The caller is responsible to release the memory (ie: calling SAVAPI3_free() on binblock).
This function is obsolete. Instead, please use the new, hex2bin() function.

SAVAPI3_EXP int bin2asc (const char * *binblock*, char ** *ascblock*)

Convert the file name to ASCII hex representation.

Parameters:

binblock [IN]: The file name to convert
ascblock [OUT]: The ASCII hex file name representation

Returns:

SAVAPI3_S_OK for success or an error code

Note:

The ascblock parameter will be internally allocated. The caller is responsible to release the memory (ie: calling SAVAPI3_free() on ascblock).
This function is obsolete. Instead, please use the new, bin2hex() function.

SAVAPI3_EXP int bin2hex (const char * *binblock*, SAVAPI_SIZE_T *binblock_size*, char * *hexblock*, SAVAPI_SIZE_T * *hexblock_size*)

Transform the string given in binblock in the equivalent encoded string in hexblock.

Parameters:

binblock [IN]: Buffer containing the string to be transformed.
binblock_size [IN]: Number of characters to encode from the binblock
hexblock [OUT]: Will contain the transformed buffer. Must be allocated by caller
hexblock_size [IN/OUT]: On input it contains the available size of the hexblock, on output will contain needed size for the resulted hexblock It must be at least two times the binblock_size (each character will be displayed as 2 hexadecimal digits)

Returns:

- SAVAPI3_S_OK for success,
- SAVAPI3_E_INVALID_PARAMETER if one of the input parameters is NULL, or the binblock length exceeds the maximum allowed buffer size
- SAVAPI3_E_BUFFER_TOO_SMALL if the given size is too small,
- SAVAPI3_E_CONVERSION_FAILED if the conversion could not be performed

Note:

If the hexblock buffer is not big enough it will return an error and will put the needed size in the hexblock_size parameter on output.
Unlike the bin2asc function, this functions does not internally allocate anything. It will use buffers allocated by the caller according to the given size.

SAVAPI3_EXP int hex2bin (const char * *hexblock*, SAVAPI_SIZE_T *hexblock_size*, char * *binblock*, SAVAPI_SIZE_T * *binblock_size*)

Converts a series of hex-encoded characters to normal binary encoding.

Parameters:

hexblock [IN]: Buffer containing an already encoded sequence of characters
hexblock_size [IN]: Number of characters to decode from the hexblock
binblock [OUT]: Contains the original string. Must be allocated by caller
binblock_size [IN/OUT]: On input contains the available size of the binblock, on output will contain needed size for the resulted binblock It must be at least half the binblock_size (each character is displayed as 2 hexadecimal digits)

Returns:

- SAVAPI3_S_OK for success,

- SAVAPI3_E_INVALID_PARAMETER if one of the input parameters is NULL, or the binblock length exceeds the maximum allowed buffer size
- SAVAPI3_E_BUFFER_TOO_SMALL if the given size is too small,
- SAVAPI3_E_CONVERSION_FAILED if the conversion could not be performed

Note:

Unlike the asc2bin function, this functions does not internally allocate anything. It will use buffers allocated by the caller according to the given size.

7.2 savapi3.h File Reference

7.2.1 Data Structures

- struct SAVAPI3_global_init
- *The structure used at SAVAPI initialization.* struct SAVAPI3_instance_init
- *The structure used at SAVAPI instance creation.* struct SAVAPI3_file_info
- *Contains data about the scanned file.* struct SAVAPI3_malware_info
- *Contains data about the found malware in an infected/suspicious file.* struct SAVAPI3_pre_scan_data
- *Contains the data sent to a prescan callback.* struct SAVAPI3_archive_open_data
- *Contains the data sent to a archive_open callback.* struct SAVAPI3_key_value
- *Generic container.* struct SAVAPI3_file_status_data
- *Contains the data sent to a report file status callback.* struct SAVAPI3_error_data
- *The structure associated with report error callback.* struct SAVAPI3_report_progress_data
- *The structure associated with report progress callback.* struct SAVAPI3_iframe_url_data
- *Structure associated with the iframe report.* struct SAVAPI3_report_content_data
- *The structure associated with report content callback.* union SAVAPI3_report_content_data::_content_data
- struct SAVAPI3_alert_url_data
- *Structure associated with the ALERTURL report.* struct SAVAPI3_repairable_data
- *Structure associated with the REPAIRABLE report.* struct SAVAPI3_report_scan_details_data
- *The structure associated with report scan details callback.* union SAVAPI3_report_scan_details_data::_scan_details_data
- struct SAVAPI3_callback_data
- *Structure passed by SAVAPI to a user defined callback, containing all the necessary data.* union SAVAPI3_callback_data::specific_data
- *Callbacks specific data.* struct SAVAPI3_signal_data
- *The structure to be passed when sending a signal.* struct SAVAPI3_command_data
- *The structure to be passed when sending a command.* struct SAVAPI3_version

7.2.2 *The structure used to retrieve SAVAPI version.* Defines

- #define SAVAPI3_S_OK 0
  Operation ended with success.
- #define SAVAPI3_E_INVALID_PARAMETER 1
  One of supplied parameters is invalid.
- #define SAVAPI3_E_ALREADY_INITIALIZED 2
  SAVAPI was already initialized.
- #define SAVAPI3_E_NOT_INITIALIZED 3
  SAVAPI is not initialized.
- #define SAVAPI3_E_BUFFER_TOO_SMALL 4
  Supplied buffer is too small.
- #define SAVAPI3_E_CONNECTION_MODE_NOT_SET 5
  Connection mode flag is not set.

- #define SAVAPI3_E_HOSTNAME_NOT_SET 6
  Host name is not set.
- #define SAVAPI3_E_NO_MEMORY 7
  Memory allocation failed.
- #define SAVAPI3_E_VDF_NOT_FOUND 8
  VDF file(s) not found.
- #define SAVAPI3_E_VDF_READ 9
  VDF file(s) read failed.
- #define SAVAPI3_E_VDF_CRC 10
  VDF file(s) crc check failed.
- #define SAVAPI3_E_VDF_VERSION 11
  Inconsistent versions in VDF files set.
- #define SAVAPI3_E_WRONG_ENGINE 12
  Engine initialization failed.
- #define SAVAPI3_E_ENGINE_NOT_FOUND 13
  Engine file(s) not found.
- #define SAVAPI3_E_WRONG_SAVAPI 14
  Invalid SAVAPI binary encountered.
- #define SAVAPI3_E_SELFCHK_PATCHED 15
  Inconsistent versions in engine files set.
- #define SAVAPI3_E_SELFCHK_FILE_ERR 16
  Engine file(s) read failed.
- #define SAVAPI3_E_SELFCHK_FILE_CRC 17
  Engine file(s) crc check failed.
- #define SAVAPI3_E_KEYFILE 18
  Keyfile error.
- #define SAVAPI3_E_INTERNAL 19
  SAVAPI internal error.
- #define SAVAPI3_E_NOT_SUPPORTED 20
  Unsupported feature.
- #define SAVAPI3_E_RESULT_ERROR 21
  An error occurred during a file scan.
- #define SAVAPI3_E_RESULT_FILE_NOT_FOUND 22
  Could not extract file.
- #define SAVAPI3_E_OPTION_NOT_SUPPORTED 23
  Option is not supported.
- #define SAVAPI3_E_HIT_MAX_REC 24
  Archive maximum recursion limit reached.
- #define SAVAPI3_E_HIT_MAX_SIZE 25
  Archive maximum extraction size reached.
- #define SAVAPI3_E_HIT_MAX_RATIO 26
  Archive maximum extraction ratio reached.
- #define SAVAPI3_E_ENCRYPTED 27
  Encrypted contents found.
- #define SAVAPI3_E_UNSUPPORTED 28
  Unsupported archive type/format.
- #define SAVAPI3_E_PROC_ERROR 29

Archive generic processing error.

- #define SAVAPI3_E_INCOMPLETE 30
  File was not completely scanned.
- #define SAVAPI3_E_PARTIAL 31
  Cannot extract multi-volume archive.
- #define SAVAPI3_E_HIT_MAX_COUNT 32
  Maximum number of files in archive reached.
- #define SAVAPI3_E_ABORTED 33
  Scan was aborted by signal.
- #define SAVAPI3_E_TIMEOUT 34
  Scan timed out.
- #define SAVAPI3_E_RESULT_SUSPICIOUS 35
  Possible infected file.
- #define SAVAPI3_E_DECRYPT 36
  Could not decrypt virus.
- #define SAVAPI3_E_SECTOR_READ 37
  Read error (boot record access).
- #define SAVAPI3_E_SECTOR_WRITE 38
  Write error (boot record access).
- #define SAVAPI3_E_SECTOR_INVALID 39
  Invalid sector (no bios signature) (boot record access).
- #define SAVAPI3_E_FILE_OPEN 40
  Could not open file.
- #define SAVAPI3_E_FILE_READ 41
  Could not read file.
- #define SAVAPI3_E_FILE_WRITE 42
  Could not write file.
- #define SAVAPI3_E_DEMOMODE 43
  SAVAPI in DEMO mode. Call not executed.
- #define SAVAPI3_E_QUERY_DISK_PARAM 44
  Problem while getting disk geometry.
- #define SAVAPI3_E_FILE_LEN 45
  Wrong file size in directory.
- #define SAVAPI3_E_FILE_DATE 46
  Invalid file date.
- #define SAVAPI3_E_FILE_DAMAGED 47
  Possible corrupted file.
- #define SAVAPI3_E_RESULT_DROPPER 48
  Macro heuristic: possible dropper.
- #define SAVAPI3_E_RESULT_TROJAN 49
  Macro heuristic: possible trojan horse.
- #define SAVAPI3_E_RESULT_POLYMORPHIC 50
  Macro heuristic: possible polymorphic virus.
- #define SAVAPI3_E_FORCE_BACKUP 51
  MBS is ok, force a backup to user.
- #define SAVAPI3_E_PARTITION_TABLE 52
  Partition tables unequal.

- #define SAVAPI3_E_RESULT_BOOTIMAGE 53
  File contains a boot virus image.
- #define SAVAPI3_E_FILE_PACKED 54
  File is packed PKLite or LZExe.
- #define SAVAPI3_E_FILE_OLE 55
  File is a compound doc (OLE2).
- #define SAVAPI3_E_FILE_TEMPLATE 56
  File contains a word template.
- #define SAVAPI3_E_FILE_MACRO 57
  File contains macros.
- #define SAVAPI3_E_FILE_ARCHIVE 58
  File is an archive.
- #define SAVAPI3_E_SECTOR_KNOWN 59
  Known good boot sector.
- #define SAVAPI3_E_SECTOR_UNKNOWN 60
  Unknown boot sector.
- #define SAVAPI3_E_SECTOR_CONSTANT 61
  Boot sector contains constant data.
- #define SAVAPI3_E_NOT_UPTODATE 62
  SAVAPI is not up to date.
- #define SAVAPI3_E_SETUP_PRODUCT 63
  SAVAPI product is not set.
- #define SAVAPI3_E_NO_PARAMETER 64
  No parameter given to command.
- #define SAVAPI3_E_INVALID_VALUE 65
  Invalid value in configuration or command.
- #define SAVAPI3_E_CHDIR_FAILED 66
  Could not change directory.
- #define SAVAPI3_E_NOT_ABSOLUTE_PATH 67
  Path is not absolute.
- #define SAVAPI3_E_DIR_NOT_EXISTS 68
  Directory path does not exist.
- #define SAVAPI3_E_MATCHED 69
  File was filtered from scanning.
- #define SAVAPI3_E_CONVERSION_FAILED 70
  Converting failed.
- #define SAVAPI3_E_FILE_OFFICE 71
  Office document found.
- #define SAVAPI3_E_FILE_IN_ARCHIVE 72
  Filename from archive.
- #define SAVAPI3_E_CONNECTION_FAILED 73
  Connection with the SAVAPI Service failed.
- #define SAVAPI3_E_RECEIVE_FAILED 74
  Failed to receive data from the SAVAPI Service.
- #define SAVAPI3_E_SEND_FAILED 75
  Failed to send data to the SAVAPI Service.
- #define SAVAPI3_E_OPTION_VALUE_INVALID 76

Invalid option value.

- #define SAVAPI3_E_REPAIR_FAILED 77
  Repair an infected file failed.
- #define SAVAPI3_E_FILE_CREATE 78
  Failed to create file.
- #define SAVAPI3_E_FILE_DELETE 79
  Failed to delete file.
- #define SAVAPI3_E_FILE_CLOSE 80
  Failed to close file.
- #define SAVAPI3_E_UNKNOWN 81
  Unknown engine error.
- #define SAVAPI3_E_PREFIX_SET 90
  Failed to set a detect type option.
- #define SAVAPI3_E_PREFIX_GET 91
  Failed to retrieve a detect type option.
- #define SAVAPI3_E_INVALID_QUERY 92
  Invalid query for SAVAPI Service.
- #define SAVAPI3_E_KEY_NO_KEYFILE 101
  Keyfile has not been found.
- #define SAVAPI3_E_KEY_ACCESS_DENIED 102
  Access to key file has been denied.
- #define SAVAPI3_E_KEY_INVALID_HEADER 103
  An invalid header has been found.
- #define SAVAPI3_E_KEY_KEYFILE_VERSION 104
  Invalid keyfile version number.
- #define SAVAPI3_E_KEY_NO_LICENSE 105
  No valid license found.
- #define SAVAPI3_E_KEY_FILE_INVALID 106
  Key file is invalid (invalid CRC).
- #define SAVAPI3_E_KEY_RECORD_INVALID 107
  Invalid license record detected.
- #define SAVAPI3_E_KEY_EVAL_VERSION 108
  Application is evaluation version.
- #define SAVAPI3_E_KEY_DEMO_VERSION 109
  Application is demo version.
- #define SAVAPI3_E_KEY_ILLEGAL_LICENSE 110
  Illegal (cracked) license in keyfile.
- #define SAVAPI3_E_KEY_NO_FUP_LICENSE 111
  No FUP II/III license found.
- #define SAVAPI3_E_KEY_NO_FUP2_KEYFILE 112
  No FUP II/III keyfile found.
- #define SAVAPI3_E_KEY_EXPIRED 113
  This key has expired.
- #define SAVAPI3_E_KEY_READ 114
  Error reading from key file.
- #define SAVAPI3_E_LICENSE_RESTRICTION 120
  Operation not allowed (license restriction).

- #define SAVAPI3_E_LOADING_ENGINE_MODULES 121
  Error loading engine modules.
- #define SAVAPI3_E_BUSY 122
  SAVAPI is busy.
- #define SAVAPI3_E_ENCRYPTED_MIME 123
  Encrypted mail found.
- #define SAVAPI3_E_NON_ADDRESSABLE 124
  Non addressable memory location.
- #define SAVAPI3_E_MEMORY_LIMIT 125
  Internal memory limit reached.
- #define SAVAPI3_E_PROC_INCOMPLETE_BLOCK_READ 150
  Incomplete archive block read.
- #define SAVAPI3_E_PROC_BAD_HEADER 151
  Bad archive header.
- #define SAVAPI3_E_PROC_INVALID_COMPRESSED_DATA 152
  Bad compressed data.
- #define SAVAPI3_E_PROC_OBSOLETE 153
  Obsolete archive information.
- #define SAVAPI3_E_PROC_BAD_FORMAT 154
  Bad header format.
- #define SAVAPI3_E_PROC_HEADER_CRC 155
  Bad header crc.
- #define SAVAPI3_E_PROC_DATA_CRC 156
  Bad data crc.
- #define SAVAPI3_E_PROC_FILE_CRC 157
  Bad crc for extracted file.
- #define SAVAPI3_E_PROC_BAD_TABLE 158
  Invalid decompression table.
- #define SAVAPI3_E_PROC_UNEXPECTED_EOF 159
  Unexpected end of file.
- #define SAVAPI3_E_PROC_ARCHIVE_HANDLE 160
  Archive internal handle error.
- #define SAVAPI3_E_PROC_NO_FILES_TO_EXTRACT 161
  No files could be extracted.
- #define SAVAPI3_E_PROC_CALLBACK 162
  Archive internal callback error.
- #define SAVAPI3_E_PROC_TOTAL_LOSS 163
  File extraction failed.
- #define SAVAPI3_ECAT_ERROR_IO 0
- #define SAVAPI3_ECAT_ERROR_SCAN 1
- #define SAVAPI3_ECAT_ERROR_UNPACK 2
- #define SAVAPI3_ECAT_ERROR_GENERIC 3
- #define SAVAPI3_ELEVEL_ERROR 0
- #define SAVAPI3_ELEVEL_WARNING 1
- #define SAVAPI3_ELEVEL_INFO 2
- #define SAVAPI3_FLAG_USE_TCP 1
- #define SAVAPI3_FLAG_USE_LOCAL_SOCKET 2
- #define SAVAPI3_W_DAMAGED 1

- #define SAVAPI3_W_OLE_DAMAGED 2
- #define SAVAPI3_W_SUSPICIOUS 4
- #define SAVAPI3_W_PROGRESS_ABORT 8
- #define SAVAPI3_W_HEADER_MALFORMED 16
- #define SAVAPI3_W_POTENTIAL_ARCH_BOMB 32
- #define SAVAPI3_W_RATIO_EXCEEDED 64
- #define SAVAPI3_W_MAX_EXTRACTED 128
- #define SAVAPI3_HTML_CONTENT_ATTRIB_INVISIBLE 1
- #define SAVAPI3_HTML_CONTENT_ATTRIB_EXTRASMALL 2
- #define SAVAPI3_HTML_CONTENT_ATTRIB_ODDPOS 4
- #define SAVAPI3_HTML_CONTENT_ATTRIB_MALICIOUS 8
- #define SAVAPI3_I_OLEFILE 1
- #define SAVAPI3_I_TEMPLATE 2
- #define SAVAPI3_I_MACROS_PRESENT 4
- #define SAVAPI3_I_ALL_MACROS_DELETED 8
- #define SAVAPI3_I_OLE_ENCRYPTED 16
- #define SAVAPI3_I_ACTIVE_CONTENT_PRESENT 32
- #define SAVAPI3_I_MAILBOX 64
- #define SAVAPI3_PUBLIC_OPTIONS 0
  Marks the start of the space used for SAVAPI public options.
- #define SAVAPI3_OPTION_CWD SAVAPI3_PUBLIC_OPTIONS + 1
  Specifies current working directory for SAVAPI.
- #define SAVAPI3_OPTION_CONF SAVAPI3_PUBLIC_OPTIONS + 2
  Specifies the configuration file that is used.
- #define SAVAPI3_OPTION_ARCHIVE_SCAN SAVAPI3_PUBLIC_OPTIONS + 3
  Activates archive detection and scanning.
- #define SAVAPI3_OPTION_ARCHIVE_MAX_SIZE SAVAPI3_PUBLIC_OPTIONS + 4
  Set the maximum allowed size (in bytes) for any file within an archive.
- #define SAVAPI3_OPTION_ARCHIVE_MAX_REC SAVAPI3_PUBLIC_OPTIONS + 5
  Set the maximum allowed recursion within an archive.
- #define SAVAPI3_OPTION_ARCHIVE_MAX_RATIO SAVAPI3_PUBLIC_OPTIONS + 6
  Set the maximum allowed decompressing-ratio within an archive.
- #define SAVAPI3_OPTION_ARCHIVE_MAX_COUNT SAVAPI3_PUBLIC_OPTIONS + 7
  Set the maximum allowed number of files within an archive.
- #define SAVAPI3_OPTION_MAILBOX_SCAN SAVAPI3_PUBLIC_OPTIONS + 8
  Activates detection and scanning of mailboxes.
- #define SAVAPI3_OPTION_HEUR_MACRO SAVAPI3_PUBLIC_OPTIONS + 9
  Activates heuristic macro detection.
- #define SAVAPI3_OPTION_HEUR_LEVEL SAVAPI3_PUBLIC_OPTIONS + 10
  Set the heuristic level for the engine. The available levels are:
  - 0 - Disable heuristic detection.
  - 1 - Lazy heuristic detection. This is the lowest possible mode, detection is not very good, but the false positives number will be low.
  - 2 - Normal heuristic detection.
  - 3 - High heuristic detection. This is the highest possible mode, but the false positives number will be high.
- #define SAVAPI3_OPTION_SCAN_TEMP SAVAPI3_PUBLIC_OPTIONS + 11
  Set the temporary directory used for scanning files.
- #define SAVAPI3_OPTION_SCAN_TIMEOUT SAVAPI3_PUBLIC_OPTIONS + 12
  Set the maximum number of seconds allowed to scan a file before aborting.

- #define SAVAPI3_OPTION_REPAIR SAVAPI3_PUBLIC_OPTIONS + 13
  Activates the repairing of infected files.
- #define SAVAPI3_OPTION_NOTIFY_REPAIR SAVAPI3_PUBLIC_OPTIONS + 14
  Activates the notification of reparable infected files.
- #define SAVAPI3_OPTION_NOTIFY_OFFICE SAVAPI3_PUBLIC_OPTIONS + 15
  Activates the detection of office documents.
- #define SAVAPI3_OPTION_NOTIFY_OFFICE_MACRO SAVAPI3_PUBLIC_OPTIONS + 16
  Activates the detection of macros within office documents.
- #define SAVAPI3_OPTION_UPDATE_SERVERS SAVAPI3_PUBLIC_OPTIONS + 17
  Specify the list of update servers.
- #define SAVAPI3_OPTION_UPDATE_PROXY SAVAPI3_PUBLIC_OPTIONS + 18
  Activate usage of a proxy server for updates.
- #define SAVAPI3_OPTION_UPDATE_PROXY_SETTINGS SAVAPI3_PUBLIC_OPTIONS + 19
  Specify the settings for a proxy server.
- #define SAVAPI3_OPTION_NOTIFY_ALERTURL SAVAPI3_PUBLIC_OPTIONS + 20
  Activates the notification of virus description url.
- #define SAVAPI3_OPTION_DETECT_ADSPY SAVAPI3_PUBLIC_OPTIONS + 21
  Activate detection for the specified type.
- #define SAVAPI3_OPTION_DETECT_APPL SAVAPI3_PUBLIC_OPTIONS + 22
  Activate detection for the specified type.
- #define SAVAPI3_OPTION_DETECT_BDC SAVAPI3_PUBLIC_OPTIONS + 23
  Activate detection for the specified type.
- #define SAVAPI3_OPTION_DETECT_DIAL SAVAPI3_PUBLIC_OPTIONS + 24
  Activate detection for the specified type.
- #define SAVAPI3_OPTION_DETECT_GAME SAVAPI3_PUBLIC_OPTIONS + 25
  Activate detection for the specified type.
- #define SAVAPI3_OPTION_DETECT_HIDDENEXT SAVAPI3_PUBLIC_OPTIONS + 26
  Activate detection for the specified type.
- #define SAVAPI3_OPTION_DETECT_JOKE SAVAPI3_PUBLIC_OPTIONS + 27
  Activate detection for the specified type.
- #define SAVAPI3_OPTION_DETECT_PCK SAVAPI3_PUBLIC_OPTIONS + 28
  Activate detection for the specified type.
- #define SAVAPI3_OPTION_DETECT_PHISH SAVAPI3_PUBLIC_OPTIONS + 29
  Activate detection for the specified type.
- #define SAVAPI3_OPTION_DETECT_SPR SAVAPI3_PUBLIC_OPTIONS + 30
  Activate detection for the specified type.
- #define SAVAPI3_OPTION_IFRAMES_URL SAVAPI3_PUBLIC_OPTIONS + 31
  Activate IFRAME detection.
- #define SAVAPI3_OPTION_REPORT_ENCRYPTED_MIME SAVAPI3_PUBLIC_OPTIONS + 32
  Activate reporting of encrypted mails.
- #define SAVAPI3_OPTION_SCAN_MODE SAVAPI3_PUBLIC_OPTIONS + 33
  Set the scanning method. Available options are:
  - SMART - Smart Extensions scan mode. The files scanned for malware are chosen by SAVAPI The choice is made based on the files content. This is the recommended setting.
  - ALL - All scan mode. Files are scanned for malware, no matter their content or extension.
  - EXTLIST - Extensions List scan mode. Only files with specific extensions are scanned for malware content.

- #define SAVAPI3_OPTION_MIME_SCAN SAVAPI3_PUBLIC_OPTIONS + 34
  Activate detection and scanning of mails.
- #define SAVAPI3_OPTION_PGP_SCAN SAVAPI3_PUBLIC_OPTIONS + 35
  Activate scanning and reporting of PGP binaries.
- #define SAVAPI3_OPTION_PRODUCT SAVAPI3_PUBLIC_OPTIONS + 40
  Set the key-id that is required by the application.
- #define SAVAPI3_OPTION_DETECT_ALLTYPES SAVAPI3_PUBLIC_OPTIONS + 41
- #define SAVAPI3_OPTION_SAVAPI SAVAPI3_PUBLIC_OPTIONS + 50
- #define SAVAPI3_OPTION_AVE_VERSION SAVAPI3_PUBLIC_OPTIONS + 51
- #define SAVAPI3_OPTION_VDF_VERSION SAVAPI3_PUBLIC_OPTIONS + 52
- #define SAVAPI3_OPTION_PID SAVAPI3_PUBLIC_OPTIONS + 53
  Retrieve the process-id for the SAVAPI process that is currently handling the TCP/IP connection.
- #define SAVAPI3_OPTION_EXPIRE SAVAPI3_PUBLIC_OPTIONS + 54
- #define SAVAPI3_OPTION_VDFSIGCOUNT SAVAPI3_PUBLIC_OPTIONS + 55
- #define SAVAPI3_OPTION_SELECTABLE_DETECT SAVAPI3_PUBLIC_OPTIONS + 56
  Retrieve the various types that can be detected (and dynamically turned on/off).
- #define SAVAPI3_OPTION_DESCR_ADSPY SAVAPI3_PUBLIC_OPTIONS + 57
- #define SAVAPI3_OPTION_DESCR_APPL SAVAPI3_PUBLIC_OPTIONS + 58
- #define SAVAPI3_OPTION_DESCR_BDC SAVAPI3_PUBLIC_OPTIONS + 59
- #define SAVAPI3_OPTION_DESCR_DIAL SAVAPI3_PUBLIC_OPTIONS + 60
- #define SAVAPI3_OPTION_DESCR_GAME SAVAPI3_PUBLIC_OPTIONS + 61
- #define SAVAPI3_OPTION_DESCR_HIDDENEXT SAVAPI3_PUBLIC_OPTIONS + 62
- #define SAVAPI3_OPTION_DESCR_JOKE SAVAPI3_PUBLIC_OPTIONS + 63
- #define SAVAPI3_OPTION_DESCR_PCK SAVAPI3_PUBLIC_OPTIONS + 64
- #define SAVAPI3_OPTION_DESCR_PHISH SAVAPI3_PUBLIC_OPTIONS + 65
- #define SAVAPI3_OPTION_DESCR_SPR SAVAPI3_PUBLIC_OPTIONS + 66
- #define SAVAPI3_OPTION_VDF_DATE SAVAPI3_PUBLIC_OPTIONS + 67
- #define SAVAPI3_CALLBACK_REPORT_FILE_STATUS 0
  Triggered after a file is scanned. The callback data contains the status of the last scanned file.
- #define SAVAPI3_CALLBACK_REPORT_ERROR 3
  Triggered to report an error or a warning.
- #define SAVAPI3_CALLBACK_PRE_SCAN 4
  Triggered before the scanning begins. Can be used to create filters. For example, if we want to scan only .exe files, we install a PRE_SCAN callback. Before each file is scanned, the PRE_SCAN callback will be called. Inside our implementation of the callback, we implement the filter. If the returned code is success, the file will be scanned, otherwise it will be skipped.
- #define SAVAPI3_CALLBACK_ARCHIVE_OPEN 5
  Triggered before opening an archive. If the returned code is success, the archive will be opened, otherwise it will be skipped from opening.
- #define SAVAPI3_CALLBACK_PROGRESS_REPORT 6
  Triggered when messages related to scan progress are available.
- #define SAVAPI3_CALLBACK_CONTENT_REPORT 7
  Triggered when messages related to scan (progress, warnings or infos that are not error_callback related) are available. IFRAME detection (SAVAPI3_OPTION_IFRAMES_URL) will be reported through this callback.
- #define SAVAPI3_CALLBACK_SCAN_DETAILS_REPORT 8
  Triggered when messages related to scan process details are available. The virus description url (SAVAPI3_OPTION_NOTIFY_ALERTURL).
- #define SAVAPI3_REPORT_ALERTURL 1
- #define SAVAPI3_REPORT_REPAIRABLE 2
- #define SAVAPI3_REPORT_CONTENT_IFRAME 0

- #define SAVAPI3_SIGNAL_SCAN_ABORT 1
  Will cause the SAVAPI instance to abort scanning process as soon as possible.
- #define SAVAPI3_COMMAND_RELOAD 0
  Synchronous commands for the SAVAPI client-mode.
- #define SAVAPI3_COMMAND_SHUTDOWN 2
- #define SAVAPI3_COMMAND_PING 3
- #define SAVAPI3_COMMAND_UPDATE_CHECK 4
- #define SAVAPI3_COMMAND_UPDATE_START 5
- #define SAVAPI3_SCAN_STATUS_CLEAN 0
- #define SAVAPI3_SCAN_STATUS_INFECTED 1
- #define SAVAPI3_SCAN_STATUS_SUSPICIOUS 2
- #define SAVAPI3_SCAN_STATUS_ERROR 3
- #define SAVAPI3_SCAN_STATUS_FINISHED 4
- #define SAVAPI3_FTYPE_REGULAR 4
- #define SAVAPI3_FTYPE_ARCHIVE 1
- #define SAVAPI3_FTYPE_IN_ARCHIVE 2

7.2.3 Typedefs

- typedef struct SAVAPI3_global_init SAVAPI3_GLOBAL_INIT
  The structure used at SAVAPI initialization.
- typedef struct SAVAPI3_instance_init SAVAPI3_INSTANCE_INIT
  The structure used at SAVAPI instance creation.
- typedef struct SAVAPI3_file_info SAVAPI3_FILE_INFO
  Contains data about the scanned file.
- typedef struct SAVAPI3_malware_info SAVAPI3_MALWARE_INFO
  Contains data about the found malware in an infected/suspicious file.
- typedef struct SAVAPI3_pre_scan_data SAVAPI3_PRESCAN_DATA
  Contains the data sent to a prescan callback.
- typedef struct SAVAPI3_archive_open_data SAVAPI3_ARCHIVE_OPEN_DATA
  Contains the data sent to a archive_open callback.
- typedef struct SAVAPI3_key_value SAVAPI3_KEY_VALUE
  Generic container.
- typedef struct SAVAPI3_file_status_data SAVAPI3_FILE_STATUS_DATA
  Contains the data sent to a report file status callback.
- typedef struct SAVAPI3_error_data SAVAPI3_ERROR_DATA
  The structure associated with report error callback.
- typedef struct SAVAPI3_report_progress_data SAVAPI3_REPORT_PROGRESS_DATA
  The structure associated with report progress callback.
- typedef struct SAVAPI3_iframe_url_data SAVAPI3_IFRAME_URL_DATA
  Structure associated with the iframe report.
- typedef struct SAVAPI3_report_content_data SAVAPI3_REPORT_CONTENT_DATA
  The structure associated with report content callback.
- typedef struct SAVAPI3_alert_url_data SAVAPI3_ALERT_URL_DATA
  Structure associated with the ALERTURL report.
- typedef struct SAVAPI3_repairable_data SAVAPI3_REPAIRABLE_DATA
  Structure associated with the REPAIRABLE report.
- typedef struct SAVAPI3_report_scan_details_data SAVAPI3_REPORT_SCAN_DETAILS_DATA
  The structure associated with report scan details callback.

- typedef struct SAVAPI3_callback_data SAVAPI3_CALLBACK_DATA
  Structure passed by SAVAPI to a user defined callback, containing all the necessary data.
- typedef struct SAVAPI3_signal_data SAVAPI3_SIGNAL_DATA
  The structure to be passed when sending a signal.
- typedef struct SAVAPI3_command_data SAVAPI3_COMMAND_DATA
  The structure to be passed when sending a command.
- typedef struct SAVAPI3_version SAVAPI3_VERSION
  The structure used to retrieve SAVAPI version.
- typedef enum _SAVAPI3_log_level SAVAPI3_LOG_LEVEL
  The enumeration used to specify the SAVAPI's logging levels.
- typedef void * SAVAPI3_FD
  SAVAPI instance handle.
- typedef int(* SAVAPI3_CALLBACK)(SAVAPI3_CALLBACK_DATA *data)
  SAVAPI callback function pointer definition.
- typedef void(* SAVAPI3_LOG_CALLBACK)(SAVAPI3_LOG_LEVEL log_level, const SAVAPI_TCHAR *message, void *user_data)
  SAVAPI callback for logging.

7.2.4 Enumerations

- enum _SAVAPI3_log_level { SAVAPI3_LOG_DEBUG = 0, SAVAPI3_LOG_INFO, SAVAPI3_LOG_WARNING, SAVAPI3_LOG_ALERT, SAVAPI3_LOG_ERROR }
  The enumeration used to specify the SAVAPI's logging levels.

7.2.5 Functions

- int SAVAPI3_EXP SAVAPI3_set_log_callback (SAVAPI3_LOG_CALLBACK log_fct, SAVAPI3_LOG_LEVEL min_level, void *user_data)
  Sets the SAVAPI logging function.
- int SAVAPI3_EXP SAVAPI3_initialize (SAVAPI3_GLOBAL_INIT *savapi_init)
  SAVAPI initialization function Initializes the SAVAPI library, according to the parameters specified in the initialization structure. It should be called once per process.
- int SAVAPI3_EXP SAVAPI3_uninitialize ()
  SAVAPI uninitialization function Uninitializes the SAVAPI library, cleaning up all used resources. Once called, all subsequent SAVAPI calls will fail with SAVAPI3_E_NOT_INITIALIZED error code.
- int SAVAPI3_EXP SAVAPI3_get_version (SAVAPI3_VERSION *version)
  Returns SAVAPI version.
- int SAVAPI3_EXP SAVAPI3_create_instance (SAVAPI3_INSTANCE_INIT *init, SAVAPI3_FD *savapi_fd)
  SAVAPI factory function The function opens a connection to the SAVAPI daemon for client-mode, or, for library mode, it creates a new SAVAPI instance.
- int SAVAPI3_EXP SAVAPI3_release_instance (SAVAPI3_FD *savapi_fd)
  Destroys a SAVAPI handler, previously created with SAVAPI3_create_instance . The function closes the connection to the SAVAPI daemon for client-mode, or, for library mode, it releases the SAVAPI instance.
- int SAVAPI3_EXP SAVAPI3_set_user_data (SAVAPI3_FD *savapi_fd, void *user_data)
  Sets user specific data. This functions sets user data that will be returned untouched as user_data member of SAVAPI3_CALLBACK_DATA structure.
- int SAVAPI3_EXP SAVAPI3_is_running ()
  Determines if the SAVAPI daemon is running The library must be initialized with the proper daemon connection parameters for this function to run correctly.
- int SAVAPI3_EXP SAVAPI3_is_running_ex (const SAVAPI_TCHAR *hostname, unsigned int port)

Determines if the SAVAPI daemon is running on the specified interface (hostname and port).

- int SAVAPI3_EXP SAVAPI3_register_callback (SAVAPI3_FD *savapi_fd, unsigned int callback_id, SAVAPI3_CALLBACK callback)
  Registers a client defined callback.
- int SAVAPI3_EXP SAVAPI3_unregister_callback (SAVAPI3_FD *savapi_fd, unsigned int callback_id, SAVAPI3_CALLBACK callback)
  Unregisters a previously registered client defined callback.
- int SAVAPI3_EXP SAVAPI3_scan (SAVAPI3_FD *savapi_fd, SAVAPI_TCHAR *file_name)
  Starts a scanning process. During the scan operation the registered callbacks may be triggered.
- int SAVAPI3_EXP SAVAPI3_set (SAVAPI3_FD *savapi_fd, unsigned int option_id, SAVAPI_TCHAR *buffer)
  Sets SAVAPI individual settings.
- int SAVAPI3_EXP SAVAPI3_get (SAVAPI3_FD *savapi_fd, unsigned int option_id, SAVAPI_TCHAR *buffer, SAVAPI_SIZE_T *buffer_size)
  Reads SAVAPI settings.
- int SAVAPI3_EXP SAVAPI3_get_dynamic_detect (SAVAPI_TCHAR *type, int *id)
  Retrieve the various types that can be detected (and dynamically turned on/off).
- int SAVAPI3_EXP SAVAPI3_send_signal (SAVAPI3_FD *savapi_fd, unsigned int signal_id, SAVAPI3_SIGNAL_DATA *data)
  Sends a signal to a specific SAVAPI instance The SAVAPI3_scan may take a long amount of time to finish scanning its target and in some situations a forced abort would be desirable. In these kind of situations, SAVAPI3_send_signal may help by sending signals to a running SAVAPI instance (SAVAPI3_SIGNAL_SCAN_ABORT for instance).
- int SAVAPI3_EXP SAVAPI3_set_fops (SAVAPI3_FD *savapi_fd, void *fops_pointer, void *fops_context)
  Specify the new fops who will be used by the engine for reading.
- void SAVAPI3_EXP SAVAPI3_free (void **ptr)
  Frees the memory space pointed to by ptr.
- int SAVAPI3_EXP SAVAPI3_reload_engine ()
  Reloads the engine from the location given at global initialization.
- int SAVAPI3_EXP SAVAPI3_reload_engine_ex (const SAVAPI3_GLOBAL_INIT *global_init)
  Reloads the engine from the specified location.

7.3 stchar.h File Reference

Conversion between SAVAPI_TCHAR and char//TCHAR.

7.3.1 Defines

- #define SAVAPI3_EXP
- #define SAVAPI_TCHAR char
- #define SAVAPI_SIZE_T unsigned int

7.3.2 Functions

- SAVAPI3_EXP int CharToSTCHAR (SAVAPI_TCHAR **pDest, const char *pSrc)
  Convert from char//TCHAR to a SAVAPI_TCHAR.
- SAVAPI3_EXP int STCHARToChar (char **pDest, const SAVAPI_TCHAR *pSrc)
  Convert from a SAVAPI_TCHAR buffer to a char//TCHAR.

7.3.3 Detailed Description

Conversion between SAVAPI_TCHAR and char//TCHAR.

UNICODE (WIN) SAVAPI_TCHAR = wchar_t(UCS2) | char(UTF-8) UNICODE (UNIX) SAVAPI_TCHAR = wchar_t(UCS2) | char(locale) ANSI (WIN + UNIX) SAVAPI_TCHAR = char(locale) | char(locale)

7.3.4 Define Documentation

#define SAVAPI3_EXP

#define SAVAPI_SIZE_T unsigned int

#define SAVAPI_TCHAR char

7.3.5 Function Documentation

SAVAPI3_EXP int CharToSTCHAR (SAVAPI_TCHAR ** *pDest*, const char * *pSrc*)

Convert from char//TCHAR to a SAVAPI_TCHAR.

Parameters:

pDest [OUT]: Pointer to a SAVAPI_TCHAR that will hold the converted buffer
pSrc [IN]: The buffer to convert.

Returns:

SAVAPI3_S_OK for success or an error code

Note:

The pDest parameter will be internally allocated. The caller is responsible to release the memory by calling SAVAPI3_free() on pDest.

SAVAPI3_EXP int STCHARToChar (char ** *pDest*, const SAVAPI_TCHAR * *pSrc*)

Convert from a SAVAPI_TCHAR buffer to a char//TCHAR.

Parameters:

pDest [OUT]: Pointer to a char that will hold the converted buffer
pSrc [IN]: Pointer to the SAVAPI_TCHAR to convert

Returns:

SAVAPI3_S_OK for success or an error code

Note:

The pDest parameter will be internally allocated. The caller is responsible to release the memory by calling SAVAPI3_free() on pDest.