

MC-SECURITY for VB 4.0 (16/32 Bit)

(v1.13)

Overview

Serialization

File Encrypt - Decrypt

String Encrypt-Decrypt

File Crc32

String Crc32

HashMD5

Registration Key

Media-ID

GZIPFileCompress / GZIPFileExpand

GZIPStringCompress / GZIPFileExpand

RUBYencrypt / RUBYdecrypt

RUBYencryptFile / RUBYdecryptFile

MC-Security Version

Revision History

New Features

Installation

Technical Support

How to register 'MC-SECURITY' on CompuServe or on Internet

License Agreement

Distribution Note

Acknowledgement

Other products

Overview

'MC-SECURITY (16/32-Bit)' is a set of routines primarily intended for developers so that they may protect programs, applications, files or datas.

Two versions are included in the same product :

- 1) MCSEC-16.DLL for use with VB 4.0 (16-Bit) under Windows 3.1x or Windows 95.
- 2) MCSEC-32.DLL for use with VB 4.0 (32-Bit) under Windows 95/Windows NT.

Revision History

See also : [New Features](#)

Version	Comments
-	
1.13	Correct a problem with serialization in the calling EXE file (same file).
1.12	<i>no revision.</i>
1.11	<i>no revision.</i>
1.10	<i>no revision.</i>
1.00	Initial release of the 'MC-SECURITY (16/32 Bit)'.

New Features

See also : [Revision History](#)

Version	Comments
-	
1.13	<i>no new features.</i>
1.12	Add the registration of my products on Internet with KAGI SHAREWARE, use REGISTER.EXE for registration.
1.11	Read the existing media ID (volume name, volume serial number, ...) from a disk under Win95/WinNT. GetMediaID
1.10	Encode a string with a password using the RUBY algorithm (7 modes). RUBYencrypt Decode a string with a password using the RUBY algorithm (7 modes). RUBYdecrypt Copy one file to an another file but with RUBY algorithm (7 modes). RUBYencryptFile Copy one file to an another file but with RUBY algorithm (7 modes). RUBYdecryptFile Compress a string into a compressed format using GZIP compression method. GZIPStringCompress Expand a compressed string into a normal format using GZIP compression method. GZIPStringExpand Compress a file into a compressed format using GZIP compression method. GZIPFileCompress Expand a compressed file into a normal format using GZIP compression method. GZIPFileExpand
1.00	Initial release of the 'MC-SECURITY (16/32 Bit)'.

Installation

The following items are discussed :

- 1) If you're working with 'MC-SECURITY' for VB 4.0 (16-Bit) under Windows 3.1x;
- 1) If you're working with 'MC-SECURITY' for VB 4.0 (16-Bit) under Windows 95/Windows NT;
- 3) If you're working with 'MC-SECURITY' for VB 4.0 (32-Bit) under Windows 95/Windows NT.

1) If you're working with 'MC-SECURITY' for VB 4.0 (16-Bit) under Windows 3.1x

You must unZIP all files in the same directory and perform the supplementary copy below.

Demonstration version :

The files MCSEC-16.DLL should be copied in your WINDOWS\SYSTEM directory.

Registered version :

The files MCSEC-16.DLL should be copied in your WINDOWS\SYSTEM directory.
The file MCSECURE.LIC should be copied in your WINDOWS directory.

Distribution note:

When you create and distribute applications that use 'MC-SECURITY', you must install :

- 1) The file MCSEC-16.DLL must be installed in the customer's Microsoft Windows WINDOWS\SYSTEM subdirectory

The setup kit included with Visual Basic provides tools that help you write setup programs that install your applications correctly.

You are not allowed to distribute 'MCSECURE.LIC' file with any application that you distribute.

2) If you're working with 'MC-SECURITY' for VB 4.0 (16-Bit) under Windows 95/Windows NT

You must unZIP all files in the same directory and perform the supplementary copy below.

Demonstration version :

The files MCSEC-16.DLL should be copied in your WIN95\SYSTEM directory.

Registered version :

The files MCSEC-16.DLL should be copied in your WIN95\SYSTEM directory.
The file MCSECURE.LIC should be copied in your WIN95 directory.

Distribution note:

When you create and distribute applications that use 'MC-SECURITY', you must install :

- 1) The file MCSEC-16.DLL must be installed in the customer's Microsoft Windows WINDOWS\SYSTEM subdirectory

The setup kit included with Visual Basic provides tools that help you write setup programs that install your applications correctly.

You are not allowed to distribute 'MCSECURE.LIC' file with any application that you distribute.

3) If you're working with 'MC-SECURITY' for VB 4.0 (32-Bit) under Windows 95/Windows NT

You must unZIP all files in the same directory and perform the supplementary copy below.

Demonstration version :

The files MCSEC-32.DLL should be copied in your WIN95\SYSTEM directory.

Registered version :

The files MCSEC-32.DLL should be copied in your WIN95\SYSTEM directory.

The file MCSECURE.LIC should be copied in your WIN95 directory.

Distribution note:

When you create and distribute applications that use 'MC-SECURITY', you must install :

- 1) The file MCSEC-32.DLL must be installed in the customer's Microsoft Windows WIN95\SYSTEM subdirectory

The setup kit included with Visual Basic provides tools that help you write setup programs that install your applications correctly.

You are not allowed to distribute 'MCSECURE.LIC' file with any application that you distribute.

Technical Support

Only registered users can receive support and update.

To receive support, you must specify your registration ID (MCSEC-16-xxx or MCSEC-32-xxx).

However, any report on any problem are the welcome.

The following information may be of help to you in streamlining your efforts to resolve any technical problems you may have with 'mcr VB/Error Handler - Tracer Profiler' product.

GPF?

If you are getting a GPF (General Protection Fault), write down the information that is displayed when the error occurs. Also, make a note of what your code was doing (in general terms.)

ISOLATE IT

Try to isolate the cause of the error. If at all possible, step through your code with F8 and F9. Try to find the one line of code that is causing the error.

SCALE IT DOWN

If at all possible, try to reproduce the problem in a small test program that you can send in. Send your test on CompuServe.

CompuServe Mail:

Name : Michaël RENARD
CIS : 100042,3646
Internet : 100042.3646@compuserve.com

Registering 'MC-SECURITY (16/32 Bit)'

[Registering 'MC-SECURITY \(16/32-Bit\)' on CompuServe](#)

- 1) [GO SWREG](#)
- 2) Choose Register Shareware.
- 3) 'MC-SECURITY (16/32 Bit)' [SWREG ID](#) is : **#8536**.

[Registering 'MC-SECURITY \(16/32-Bit\)' on Internet](#)

- 1) Use the program REGISTER.EXE
- 2) Select the product MC-SECURITY (16/32-Bit).
- 3) Send by e-mail or fax or postal.

As soon as I receive notification of your registration (usually 1 - 3 days) I will send you out via e-Mail a license file for each single user license that you've asked, or site license or world license.

You also qualify to receive new versions of 'MC-SECURITY (16/32 Bit)' during one year.

[The price for 'MC-SECURITY \(16/32 Bit\)' is fixed at \\$10.00.](#)

This price is much a contribution to my works that a payment. When you register "MC-SECURITY (16/32-Bit)", you help me to develop better products and others products.

The 16-Bit part of 'MC/SECURITY' is written in C and compiled using Visual C++ 1.52c (maximize speed and 80386).
The 32-Bit part of 'MC/SECURITY' is written in C and compiled using Visual C++ 4.00 (maximize speed and 80486).

License Agreement

The 'MC-SECURITY' program and its associated DLL is not public domain software or free software.

The 'MC-SECURITY' program and its associated DLL is copyrighted, and all rights are reserved by its author: Michaël Renard.

You are licensed to use this software on a restricted number of computers. You may copy the software to facilitate your use of it on as many computers as there are licensed users specified in the 'MC-SECURITY' license file '**MCSECURE.LIC**'. Making copies for any other purpose violates international copyright laws.

*You are not allowed to distribute '**MCSECURE.LIC**' file with any application that you distribute.*

Disclaimer:

This software is sold AS IS without warranty of any kind, either expressed or implied, including but not limited to the implied warranties of merchantability and fitness for a particular purpose. The authors assume no liability for any alleged or actual damages arising from the use of this software. (Some states do not allow the exclusion of implied warranties, so the exclusion may not apply to you.)

Your use of this product indicates that you have read and agreed to these terms.

Distribution Note

When you create and distribute applications that use 'MC-SECURITY', you must install :

1) The files MCSEC-16.DLL or MCSEC-32.DLL must be installed in the customer's Microsoft Windows WINDOWS\SYSTEM or WIN95\SYSTEM subdirectory

The setup kit included with Visual Basic provides tools that help you write setup programs that install your applications correctly.

You are not allowed to distribute 'MCSECURE.LIC' file with any application that you distribute.

Acknowledgement

Some routines have been written by the following great developers (I've adapted their routines for VB 3.0/4.0 under Win3.1x/Win95/WinNT) :

Special thanks to [Brian Pirie](#) for [REGISTRATION KEY SYSTEM FOR C PROGRAMMERS](#).

Special thanks to [Andy Brown](#) for [MD5 HASH ALGORITHM](#). (derived from the RSA ** ** Data Security, Inc. MD5 Message-Digest Algorithm).

Special thanks to [Michael Paul Johnson](#) for [RUBY Mark 5 Algorithm](#).

About *RUBY MARK 5* algorithm, you can reach Michael Paul Johnson at :

Internet e-mail : mpj@csn.net

Web site : <http://www.csn.net/~mpj> <- Get John (GLW)

Colorado Catacombs BBS : 303-772-1062

Special thanks to [Jean-loup Gailly](#) and [Mark Adler](#) for [ZLIB](#) library.

This help has been written by using [ForeHelp v1.04](#) from [ForeFront, Inc.](#)

Other products

Basis products :

1) [TIME TO WIN \(VB 4.0 \(32-Bit\)\)](#)

This product is a powerfull 32-Bit DLL with more than 665 routines for VB 4.0 (32-Bit) application.

You can download the full product (demo included) from the following site :

CompuServe : GO MSBASIC : select T2WIN-32.ZIP
Internet : ftp.winsite.com/pub/pc/win95/programr/vbasic/T2WIN-32.ZIP

You can register the full product (demo included) at the price of **\$52.00** from the following site :

CompuServe : GO SWREG : select the product item #7516
Internet : use REGISTER.EXE in the ZIP'ed file : select [TIME TO WIN \(32-Bit\)](#)

2) [TIME TO WIN \(VB 3.0 or VB 4.0 \(16-Bit\)\)](#)

This product is a powerfull 16-Bit DLL with more than 650 routines for VB 3.0 and VB 4.0 (16-Bit) application.

You can download the full product (demo included) from the following site :

CompuServe : GO MSBASIC : select TIME2WIN.ZIP and T2WIN-16.ZIP
Internet : ftp.winsite.com/pub/pc/win3/programr/vbasic/TIME2WIN.ZIP
ftp.winsite.com/pub/pc/win3/programr/vbasic/T2WIN-16.ZIP

You can register the full product (demo included) at the price of **\$43.00** from the following site :

CompuServe : GO SWREG : select the product item #4045
Internet : use REGISTER.EXE in the ZIP'ed file : select [TIME TO WIN \(16-Bit\)](#)

3) [TIME TO WIN for MS Office 95](#)

This product is a powerfull 32-Bit DLL with more than 360 routines for Access 95, Excel 95 and Word 95.

You can download the full product (demo included) from the following site :

CompuServe : GO MSBASIC : select T2WOFFIC.ZIP
Internet : ftp.winsite.com/pub/pc/win95/programr/vbasic/T2WOFFIC.ZIP

You can register the full product (demo included) at the price of **\$25.00** from the following site :

CompuServe : GO SWREG : select the product item #10355
Internet : use REGISTER.EXE in the ZIP'ed file : select [TIME TO WIN for MS Office 95](#)

4) [mcr VB/Error Handler - Tracer Profiler](#)

This product is a powerfull product for adding/removing the management of errors and tracer-profiler for project under VB 3.0, VB 4.0 (16-Bit) and VB 4.0 (32-Bit).

You can download the full product (demo included) from the following site :

CompuServe : GO MSBASIC : select MCVBEHTP.ZIP
Internet : ftp.winsite.com/pub/pc/win95/programr/vbasic/MCVBEHTP.ZIP

You can register the full product (demo included) at the price of **\$25.00** from the following site :

CompuServe : GO SWREG : select the product item #4380
Internet : use REGISTER.EXE in the ZIP'ed file : select [MCVBEHTP for UNregistered user ...](#)

5) [MC SECURITY for VB 4.0 \(16/32 Bit\)](#)

This product is a powerfull 16/32-Bit DLL with 28 routines for VB 4.0 (16/32 Bit) application.

You can download the full product (demo included) from the following site :

CompuServe : GO MSBASIC : select MCSECURE.ZIP
Internet : ftp.winsite.com/pub/pc/win95/programr/vbasic/MCSECURE.ZIP

You can register the full product (demo included) at the price of **\$10.00** from the following site :

CompuServe : GO SWREG : select the product item #8536
Internet : use REGISTER.EXE in the ZIP'ed file : select [MC-SECURITY \(16/32-Bit\)](#)

6) [MC STRING for VB 4.0 \(32 Bit\)](#)

This product is a powerfull 32-Bit DLL with 64 routines for VB 4.0 (32 Bit) application.

You can download the full product (demo included) from the following site :

CompuServe : GO MSBASIC : select MCSTR-32.ZIP

Internet : ftp.winsite.com/pub/pc/win95/programr/vbasic/MCSTR-32.ZIP

You can register the full product (demo included) at the price of \$**10.00** from the following site :

CompuServe : GO SWREG : select the product item #12012

Internet : use REGISTER.EXE in the ZIP'ed file : select [MC-STRING \(32-Bit\)](#)

7) [MC DISK VB 4.0 \(32 Bit\)](#)

This product is a powerfull 32-Bit DLL with 94 routines for VB 4.0 (32 Bit) application.

You can download the full product (demo included) from the following site :

CompuServe : GO MSBASIC : select MCDSK-32.ZIP

Internet : ftp.winsite.com/pub/pc/win95/programr/vbasic/MCDSK-32.ZIP

You can register the full product (demo included) at the price of \$**10.00** from the following site :

CompuServe : GO SWREG : select the product item #12011

Internet : use REGISTER.EXE in the ZIP'ed file : select [MC-DISK \(32-Bit\)](#)

Update products :

1) [Update TIME TO WIN \(VB 3.0 or VB 4.0 \(16-Bit\)\) -> TIME TO WIN 32-Bit \(VB 4.0 \(32-Bit\)\)](#)

This product is an update for registered user of 'TIME TO WIN' which want register the 'TIME TO WIN (32-Bit)'.

You can download the full product (demo included) from the following site :

CompuServe : GO MSBASIC : select T2WIN-32.ZIP

Internet : ftp.winsite.com/pub/pc/win95/programr/vbasic/T2WIN-32.ZIP

You can register the full product (demo included) at the price of \$**29.00** from the following site :

CompuServe : GO SWREG : select the product item #7517

Internet : use REGISTER.EXE in the ZIP'ed file : select [update TIME TO WIN \(16-Bit\) to TIME TO WIN \(32-Bit\)](#)

Special price for registered user :

1) [If you're a registered user of 'TIME TO WIN' or 'TIME TO WIN \(32-Bit\)](#)

You receive a special price for 'mcr VB/Error Handler - Tracer Profiler' under VB 3.0, VB 4.0 (16-Bit) and VB 4.0 (32-Bit).

You can download the full product (demo included) from the following site :

CompuServe : GO MSBASIC : select MCVBEHTP.ZIP

Internet : ftp.winsite.com/pub/pc/win95/programr/vbasic/MCVBEHTP.ZIP

You can register the full product (demo included) at the price of \$**16.00** from the following site :

CompuServe : GO SWREG : select the product item #4379

Internet : use REGISTER.EXE in the ZIP'ed file : select [MCVBEHTP for registered user ...](#)

RUBYencrypt, RUBYdecrypt

Purpose :

[RUBYencrypt](#) encode a string with a password using the RUBY algorithm (7 modes).

[RUBYdecrypt](#) decode a string with a password using the RUBY algorithm (7 modes).

Declare Syntax :

For VB 4.0 (16-Bit)

Declare Function cRUBYencrypt Lib "mcsec-16.dll" (Text As String, Key As String, ByVal Mode As Integer) As String

Declare Function cRUBYdecrypt Lib "mcsec-16.dll" (Text As String, Key As String, ByVal Mode As Integer) As String

For VB 4.0 (32-Bit)

Declare Function cRUBYencrypt Lib "mcsec-32.dll" (Text As String, Key As String, ByVal Mode As Integer) As String

Declare Function cRUBYdecrypt Lib "mcsec-32.dll" (Text As String, Key As String, ByVal Mode As Integer) As String

Call Syntax :

testE = cRUBYencrypt(Text, Key)

testD = cRUBYdecrypt(Text, Key)

Where :

Text	is the string to encrypt/decrypt	
Key	is the key to use for encryption/decryption	
Mode	Public Const RUBY_MODE_MINIMUM = 1	' speed is of the essence, security
secondary.	Public Const RUBY_MODE_DESK_LOCK = 2	' reasonable compromise of speed vs
security.	Public Const RUBY_MODE_DEAD_BOLT = 4	' default = probably good enough for most
things.	Public Const RUBY_MODE_PORTABLE_SAFE = 5	' security is more important than speed.
	Public Const RUBY_MODE_ANCHORED_SAFE = 8	' speed isn't much of a concern.
	Public Const RUBY_MODE_BANK_VAULT = 10	' your pentium has nothing better to do,
anyway.	Public Const RUBY_MODE_FORT_KNOX = 16	' be cool.
test	is the string encrypted/decrypted	

Comments :

The Key is case sensitive.

The length of Text can be any size.

The length of Key must be greater or equal to 6 characters.

Examples :

```
Dim Text As String
Dim Key As String
Dim Enc As String
Dim Dec As String
```

Text = "Under the blue sky, the sun is yellow"

Key = "a new encryption"

Enc = cRUBYencrypt(Text, Key, RUBY_MODE_DESK_LOCK)

Dec = cRUBYdecrypt(Enc, Key, RUBY_MODE_DESK_LOCK)

See also :

IsSerial, SerialGet, SerialInc, SerialPut, SerialRmv

Purpose :

Serialization is a set of routines primarily intended for developers so that they may append a serial number (or other identifier) to the end of an .exe, .dll or any static files in size, put/modify or get serial numbers or any string to 50 characters. Users may use to initialize purchased software applications with ownership, security-related, or other identifying marks.

A unique serial number going out with each copy of an application affords the developer with a possible opportunity to identify, if need be, the registered client of a particular copy. The end-user is normally unaware of the existence of such a mark, its location, its method of placement or the method of reading/verifying. Its absence or modification may provide evidence of tampering.

IsSerial checks if a file has been serialized.

SerialGet gets the serialization information from a serialized file.

SerialInc increment by a value the serialized number part of a serialized file.

SerialPut puts a serialization information to a serialized file.

SerialRmv removes the serialization information from a serialized file.

Type declaration :

For VB 4.0 (16-Bit)

```
Type tagSERIALDATA16
    Description1      As String * 50      'serialization description 1
    Description2      As String * 50      'serialization description 2
    Number            As Long             'serialization number
    dummy             As String * 50      'reserved for future use
End Type
```

For VB 4.0 (32-Bit)

```
Type tagSERIALDATA32
    Description1      As String * 52      'serialization description 1
    Description2      As String * 52      'serialization description 2
    Number            As Long             'serialization number
    dummy             As String * 52      'reserved for future use
End Type
```

Constant declaration :

```
Public Const SD_SERIAL_NOT_FOUND = 1
Public Const SD_CAN_NOT_OPEN_FILE = 2
```

Declare Syntax :

For VB 4.0 (16-Bit)

```
Declare Function clsSerial Lib "mcsec-16.dll" (ByVal File As String) As Integer
Declare Function cSerialGet Lib "mcsec-16.dll" (ByVal file As String, SERIALDATA As tagSERIALDATA16) As Integer
Declare Function cSerialInc Lib "mcsec-16.dll" (ByVal file As String, ByVal Increment As Long) As Integer
Declare Function cSerialPut Lib "mcsec-16.dll" (ByVal file As String, SERIALDATA As tagSERIALDATA16) As Integer
Declare Function cSerialRmv Lib "mcsec-16.dll" (ByVal File As String) As Integer
```

For VB 4.0 (32-Bit)

```
Declare Function clsSerial Lib "mcsec-32.dll" (ByVal File As String) As Integer
Declare Function cSerialGet Lib "mcsec-32.dll" (ByVal file As String, SERIALDATA As tagSERIALDATA32) As Integer
```



```

Declare Function cSerialInc Lib "mcsec-32.dll" (ByVal file As String, ByVal Increment As Long) As Integer
Declare Function cSerialPut Lib "mcsec-32.dll" (ByVal file As String, SERIALDATA As tagSERIALDATA32) As Integer
Declare Function cSerialRmv Lib "mcsec-32.dll" (ByVal File As String) As Integer

```

Call Syntax :

```

Test% = clsSerial(File$)
Test% = cSerialGet(File$, SERIALDATA)
Test% = cSerialInc(File$, Increment&)
Test% = cSerialPut(File$, SERIALDATA)
Test% = cSerialRmv(File$)

```

Where :

File\$	is the specified file.
SERIALDATA	is a type'd variable (tagSERIALDATA).
Increment&	is the increment (positive or negative).
Test%	TRUE if all is ok, <> TRUE if an error has occurred.

Comments :

For 16-Bit :

The length of the serialization string is maximum 50 characters (SERIALDATA.Description1, SERIALDATA.Description2).

You can add/remove the serialization on the same EXE file for application. In other words, the EXE file corresponding to an application can add/remove the serialization on itself.

For 32-Bit :

The length of the serialization string is maximum 52 characters (SERIALDATA.Description1, SERIALDATA.Description2).

Due to some limitations (or some protections) in Win95/WinNT, you can't add/remove the serialization on the same EXE file for application. In other words, the EXE file corresponding to an application can't add/remove the serialization on itself.

For SerialInc :

If you pass a 0 value, the serialization number is reset to 0 (be care).

Examples :

```

Dim putSERIALDATA      As tagSERIALDATA32
Dim getSERIALDATA      As tagSERIALDATA32

putSERIALDATA.Description1 = "1234567890123456789012345"
putSERIALDATA.Description2 = ""
putSERIALDATA.Number = 987654321
Debug.Print cSerialPut("c:\tmp\sample.exe", putSERIALDATA)
Debug.Print cSerialGet("c:\tmp\sample.exe", getSERIALDATA)
Debug.Print getSERIALDATA.Description1 & Chr$(13) & getSERIALDATA.Description2 & Chr$(13) &
getSERIALDATA.Number

putSERIALDATA.Description2 = "ABCDEFGHJKLMNOPQRSTUVWXYZ"
putSERIALDATA.Number = 123456789
Debug.Print cSerialPut("c:\tmp\sample.exe", putSERIALDATA)
Debug.Print cSerialGet("c:\tmp\sample.exe", getSERIALDATA)

```

```
Debug.Print getSERIALDATA.Description1 & Chr$(13) & getSERIALDATA.Description2 & Chr$(13) &  
getSERIALDATA.Number
```

```
Debug.Print cSerialInc("c:\tmp\sample.exe", 123)  
Debug.Print cSerialGet("c:\tmp\sample.exe", getSERIALDATA)  
Debug.Print getSERIALDATA.Description1 & Chr$(13) & getSERIALDATA.Description2 & Chr$(13) &  
getSERIALDATA.Number
```

```
Debug.Print cSerialRmv("c:\tmp\sample.exe")
```

See also :

FileEncrypt, FileDecrypt

Purpose :

FileEncrypt copies one file to an another file but with encryption.

FileDecrypt copies one file to an another file but with decryption.

Constant declaration :

```
Public Const ENCRYPT_LEVEL_0 = 0
```

```
Public Const ENCRYPT_LEVEL_1 = 1
```

```
Public Const ENCRYPT_LEVEL_2 = 2
```

```
Public Const ENCRYPT_LEVEL_3 = 3
```

```
Public Const ENCRYPT_LEVEL_4 = 4
```

Declare Syntax :

For VB 4.0 (16-Bit)

```
Declare Function cFileEncrypt Lib "mcsec-16.dll" (ByVal file1 As String, ByVal file2 As String, Password As String, ByVal Level As Integer) As Long
```

```
Declare Function cFileDecrypt Lib "mcsec-16.dll" (ByVal file1 As String, ByVal file2 As String, Password As String, ByVal Level As Integer) As Long
```

For VB 4.0 (32-Bit)

```
Declare Function cFileEncrypt Lib "mcsec-32.dll" (ByVal file1 As String, ByVal file2 As String, Password As String, ByVal Level As Integer) As Long
```

```
Declare Function cFileDecrypt Lib "mcsec-32.dll" (ByVal file1 As String, ByVal file2 As String, Password As String, ByVal Level As Integer) As Long
```

Call Syntax :

```
test& = cFileEncrypt(file1, file2, password, level)
```

```
test& = cFileDecrypt(file1, file2, password, level)
```

Where :

file1\$	is the source file.
file2\$	is the destination file.
password	is the key to use for encryption/decryption.
level	level of the encryption/decryption.
test&	> 0 if all is OK (the returned value is the total bytes copied), < 0 if an error has occurred.

Comments :

The password/key is case sensitive.

The level is a number between **0** and **4** (ENCRYPT_LEVEL_0 and ENCRYPT_LEVEL_4).

Higher is the level, better is the encryption.

You must use the same level for encrypt/decrypt a gived string.

The returned value can be negative and have the following value :

-1	the password is an EMPTY string.
-32720	the number of chars in a block for writing differs from the number of chars for reading.
-32730	reading error for file 1.
-32740	writing error for file 2.
-32750	opening error for file 1.
-32751	opening error for file 2.

- 32760 allocation error for memory buffer 1.
- 32761 allocation error for memory buffer 2.

Examples :

```
test& = cFileEncrypt("c:\autoexec.bat", "c:\autoexec.tb1", "Time To Win", ENCRYPT_LEVEL_4)
test& = cFileDecrypt("c:\autoexec.tb1", "c:\autoexec.tb2", "Time To Win", ENCRYPT_LEVEL_4)
```

See also :

FileCRC32

Purpose :

FileCRC32 calculates a 32 bits CRC for a gived file.

Constant declaration :

```
Public Const OPEN_MODE_BINARY = 0
Public Const OPEN_MODE_TEXT = 1
```

Declare Syntax :

For VB 4.0 (16-Bit)

```
Declare Function cFileCRC32 Lib "mcsec-16.dll" (ByVal lpFilename As String, ByVal mode As Integer) As Long
```

For VB 4.0 (32-Bit)

```
Declare Function cFileCRC32 Lib "mcsec-32.dll" (ByVal lpFilename As String, ByVal mode As Integer) As Long
```

Call Syntax :

```
test = cFileCRC32(lpFilename, mode)
```

Where :

lpFilename	the file to proceed
mode	OPEN_MODE_BINARY (calculates the CRC on the full length of the file). This is the default mode. OPEN_MODE_TEXT (calculates the CRC until a EOF is encountered)
test	the calculated CRC 32 bits in a LONG.

Comments :

The returned value can be negative and have only a value :

-1 If the filename is not a good filename or if the filename not exist or if an error occurs when accessing the filename.

Examples :

```
test = cFileCRC32("C:\COMMAND.COM")    &h1131ADD3            (MS-DOS 6.22)
```

See also :

HashMD5

Purpose :

Performs the hash algorithm (MD5) to a specified string.

Declare Syntax :

For VB 4.0 (16-Bit)

Declare Function cHashMD5 Lib "mcsec-16.dll" (Text As String) As String

For VB 4.0 (32-Bit)

Declare Function cHashMD5 Lib "mcsec-32.dll" (Text As String) As String

Call Syntax :

Hash\$ = cHashMD5(Text\$)

Where :

Text\$	the specified string (length between 1 to 32767).
Hash\$	the returned hashed string.

Comments :

A hash algorithm such as MD5 is often used in cryptosystems to "reduce" a user-supplied passphrase into a sufficient number of bits to use as a key to the system. The following is taken from the Executive Summary section of the Internet RFC that proposes MD5 as a standard.

The [MD5] algorithm takes as input an input message of arbitrary length and produces as output a 128-bit "fingerprint" or "message digest" of the input. It is conjectured that it is computationally infeasible to produce two messages having the same message digest, or to produce any message having a given prespecified target message digest. The MD5 algorithm is intended for digital signature applications, where a large file must be "compressed" in a secure manner before being encrypted with a private (secret) key under a public-key cryptosystem such as RSA. (Source from Andy Brown).

HashMD5 is derived from the RSA ** ** Data Security, Inc. MD5 Message-Digest Algorithm.

Examples :

Dim Hash As String

```
Hash = cHashMD5("TIME TO WIN")
      -> $Ei"é£,%~"3ïXA'
```

See also :

DOSGetMediaID, DOSSetMediaID, GetMediaID

Purpose :

DOSGetMediaID read the media ID (serial number, volume label, ...) from a disk.

DOSSetMediaID change the existing media ID (serial number, volume label, ...) from a disk.

GetMediaID read the existing media ID (volume name, volume serial number, system name, ...) from a disk under Win95/WinNT.

Type declaration :

For VB 4.0 (16-Bit)

```
Type tagMEDIAID16
    InfoLevel           As Integer
    SerialNumber        As Long
    VolLabel            As String * 11
    FileSysType         As String * 8
End Type
```

For VB 4.0 (32-Bit)

```
Type tagDOSMEDIAID32
    InfoLevel           As String * 2      'use cCVI for integer conversion
    SerialNumber        As String * 4      'use cCVL for long conversion
    VolLabel            As String * 11
    FileSysType         As String * 8
End Type
```

' structure for get Media ID (WinNT)

```
Type tagMEDIAID
    VolumeName          As String
    VolumeSerialNumber  As Long
    SystemName          As String
    MaxNameLength       As Long
    FileSystemFlags     As Long
End Type
```

Declare Syntax :

For VB 4.0 (16-Bit)

```
Declare Function cDOSGetMediaID Lib "mcsec-16.dll" (ByVal nDrive As String, DOSMEDIAID As tagDOSMEDIAID16) As Integer
Declare Function cDOSSetMediaID Lib "mcsec-16.dll" (ByVal nDrive As String, DOSMEDIAID As tagDOSMEDIAID16) As Integer
```

For VB 4.0 (32-Bit)

```
Declare Function cDOSGetMediaID Lib "mcsec-32.dll" (ByVal nDrive As String, DOSMEDIAID As tagDOSMEDIAID32) As Integer
Declare Function cDOSSetMediaID Lib "mcsec-32.dll" (ByVal nDrive As String, DOSMEDIAID As tagDOSMEDIAID32) As Integer
Declare Function cGetMediaID Lib "mcsec-32.dll" (ByVal nDrive As String, MEDIAID As tagMEDIAID) As Integer
```

Call Syntax :

```
Test% = cDOSGetMediaID(nDrive$, DOSMEDIAID)
```

```
Test% = cDOSSetMediaID(nDrive$, DOSMEDIAID)
Test% = cGetMediaID(nDrive$, MEDIAID)
```

Where :

nDrive\$	is the drive letter.
DOSMEDIAID	is the type'd variable to access the drive.
MEDIAID	is the type'd variable to access the drive under WinNT.
Test%	TRUE, all is ok FALSE, no media ID or an error has occurred.

Comments :

If nDrive is empty, the default drive is used.
The informations returned by these routines are different from the GetMediaID and SetMediaID.
To decode the 'InfoLevel', you must use cCVI function.
To decode the 'SerialNumber', you must use the cCVL function.

Examples :

```
Dim DOSMEDIAID As tagDOSMEDIAID32
Dim MEDIAID As tagMEDIAID
```

```
test% = cDOSGetMediaID("A", DOSMEDIAID)
```

Drive A : no media id

```
test% = cDOSGetMediaID("B", DOSMEDIAID)
```

Drive B : no media id

```
test% = cDOSGetMediaID("C", DOSMEDIAID)
```

Drive C :	
InfoLevel : '0'	(Hex\$(cCVI(DOSMEDIAID.InfoLevel))
SerialNumber : '43361ECF'	(Hex\$(cCVL(DOSMEDIAID.SerialNumber))
VolLabel : 'UNICORN_7'	
FileSysType : 'FAT16'	

```
test% = cGetMediaID("C", MEDIAID)
```

Drive C :	
VolumeName : 'MCR'	
VolumeSerialNumber : '43361ECF'	
SystemName : 'FAT'	
MaxNameLength : '255'	
FileSystemFlags : '16390'	

See also :

GetVersion

Purpose :

GetVersion returns the version number of 'MC-SECURITY (16/32-Bit)'

Declare Syntax :

For VB 4.0 (16-Bit)

Declare Function cGetVersion Lib "mcsec-16.dll" () As Single

For VB 4.0 (32-Bit)

Declare Function cGetVersion Lib "mcsec-32.dll" () As Single

Call Syntax :

version% = cGetVersion()

Where :

Comments :

This is usefull to avoid version conflict with old version.

Examples :

version% = cGetVersion() 1.10

See also :

String Encrypt - Decrypt

Purpose :

Encrypt encodes a string with a password/key.
Decrypt decodes a string encoded with Encrypt function.

Constant declaration :

```
Public Const ENCRYPT_LEVEL_0 = 0
Public Const ENCRYPT_LEVEL_1 = 1
Public Const ENCRYPT_LEVEL_2 = 2
Public Const ENCRYPT_LEVEL_3 = 3
Public Const ENCRYPT_LEVEL_4 = 4
```

Declare Syntax :

For VB 4.0 (16-Bit)

```
Declare Function cEncrypt Lib "mcsec-16.dll" (Txt As String, password As String, ByVal level As Integer) As String
Declare Function cDecrypt Lib "mcsec-16.dll" (Txt As String, password As String, ByVal level As Integer) As String
```

For VB 4.0 (32-Bit)

```
Declare Function cEncrypt Lib "mcsec-32.dll" (Txt As String, password As String, ByVal level As Integer) As String
Declare Function cDecrypt Lib "mcsec-32.dll" (Txt As String, password As String, ByVal level As Integer) As String
```

Call Syntax :

```
test = cEncrypt(Txt, password, level)
```

Where :

Txt	is the string to encrypt
password	is the key to use for encryption
level	level of the encryption
test	is the string decrypted

Comments :

The password/key is case sensitive.
The level is a number between **0** and **4** (ENCRYPT_LEVEL_0 and ENCRYPT_LEVEL_4).
Higher is the level, better is the encryption.
You must use the same level for encrypt/decrypt a given string.

Examples :

```
Txt = "Under the blue sky, the sun is yellow"
password = "a new encryption"
level = ENCRYPT_LEVEL_4
test = cEncrypt(Txt, password, level)
txt = cDecrypt(test, password, level)
```

See also :

StringCRC32

Purpose :

StringCRC32 calculates a 32 bits CRC for a given string.

Declare Syntax :

For VB 4.0 (16-Bit)

Declare Function cStringCRC32 Lib "mcsec-16.dll" (Txt As String) As Long

For VB 4.0 (32-Bit)

Declare Function cStringCRC32 Lib "mcsec-32.dll" (Txt As String) As Long

Call Syntax :

test = cStringCRC32(Txt)

Where :

Txt	the string to proceed
test	the calculated CRC 32 bits in a LONG.

Comments :

if the string is empty, the return value is always -1 (&hFFFFFFF).

Examples :

test = cStringCRC32("ABCDEFGH")	&hE6F94BC
test = cStringCRC32("GFEDCBA")	&hF0EC0AB3

See also :

RegistrationKey

Purpose :

RegistrationKey performs the calculation of a key from a name and a code.

Declare Syntax :

For VB 4.0 (16-Bit)

Declare Function cRegistrationKey Lib "mcsec-16.dll" (ByVal RegText As String, ByVal RegCode As Long) As Long

For VB 4.0 (32-Bit)

Declare Function cRegistrationKey Lib "mcsec-32.dll" (ByVal RegText As String, ByVal RegCode As Long) As Long

Call Syntax :

Key& = cRegistrationKey(RegText\$, RegCode&)

Where :

RegText\$	the name for the registration.
RegCode&	the basis code for generating the registration
Key&	= 0, if length of RegText is < 10 or if RegKey1 is 0, <>0, the key calculated from RegText and RegKey1.

Comments :

Using this registration key system, you can easily and quickly generate and verify the validity of numerical registration keys that correspond to a person who has purchased your program. Thus, when someone who already has a shareware or demo version of your program wishes to purchase the program, you need only send them a simple registration key number, instead of sending an entire registered version. You can simply use this package to generate a unique registration key number which corresponds to the user's name (or any other string you wish to use). The user will then be able to enter this number into your software's configuration file / configuration program. When your program begins, it will be able to read this number from the configuration file, and again using this package, determine whether it is a valid registration key corresponding to the user's name. If the registration key is valid, your program can switch into "registered mode", and if not, can run in its unregistered "unregistered mode". (Source from Brian Pirie).

Examples :

```
Dim Key          As Long
Dim RegText      As String
```

```
RegText = "this is a testthis is a test"
```

```
Key = cRegistrationKey(Tmp, 123456789)
      -> 590573797
```

See also :

RUBYencryptFile, RUBYdecryptFile

Purpose :

[RUBYencryptFile](#) copy one file to an another file but with RUBY algorithm (7 modes).

[RUBYdecryptFile](#) copy one file to an another file but with RUBY algorithm (7 modes).

Declare Syntax :

For VB 4.0 (16-Bit)

Declare Function cRUBYencryptFile Lib "mcsec-16.dll" (ByVal FileIn As String, ByVal FileOut As String, Key As String, ByVal Mode As Integer) As Long

Declare Function cRUBYdecryptFile Lib "mcsec-16.dll" (ByVal FileIn As String, ByVal FileOut As String, Key As String, ByVal Mode As Integer) As Long

For VB 4.0 (32-Bit)

Declare Function cRUBYencryptFile Lib "mcsec-32.dll" (ByVal FileIn As String, ByVal FileOut As String, Key As String, ByVal Mode As Integer) As Long

Declare Function cRUBYdecryptFile Lib "mcsec-32.dll" (ByVal FileIn As String, ByVal FileOut As String, Key As String, ByVal Mode As Integer) As Long

Call Syntax :

test& = cRUBYencryptFile(FileIn, FileOut, Key, Mode)

test& = cRUBYdecryptFile(FileIn, FileOut, Key, Mode)

Where :

FileIn\$	is the source file.	
FileOut\$	is the destination file.	
Key	is the key to use for encryption/decryption.	
Mode	Public Const RUBY_MODE_MINIMUM = 1	' speed is of the essence, security
secondary.		
	Public Const RUBY_MODE_DESK_LOCK = 2	' reasonable compromise of speed vs
security.		
	Public Const RUBY_MODE_DEAD_BOLT = 4	' default = probably good enough for most
things.		
	Public Const RUBY_MODE_PORTABLE_SAFE = 5	' security is more important than speed.
	Public Const RUBY_MODE_ANCHORED_SAFE = 8	' speed isn't much of a concern.
	Public Const RUBY_MODE_BANK_VAULT = 10	' your pentium has nothing better to do,
anyway.		
	Public Const RUBY_MODE_FORT_KNOX = 16	' be cool.
test&	> 0 if all is OK (the returned value is the total bytes copied),	
	< 0 if an error has occurred.	

Comments :

The Key is case sensitive.

The length of Text can be any size.

The length of Key must be greater or equal to 6 characters.

If the returned code is a negative value, it take the following value :

```
Public Const CRYPTO_KEY_TOO_SMALL = -1
Public Const CRYPTO_CANT_INIT_KEY = -2
Public Const CRYPTO_CANT_INIT_BUFFER = -11
Public Const CRYPTO_CANT_OPEN_FILEIN = -21
Public Const CRYPTO_CANT_CREATE_FILEOUT = -22
```

```
Public Const CRYPTO_ERROR_READING_FILEIN = -31
Public Const CRYPTO_ERROR1_WRITING_FILEOUT = -41
Public Const CRYPTO_ERROR2_WRITING_FILEOUT = -42
Public Const CRYPTO_ERROR1_WRITING_LASTBYTE = -51
Public Const CRYPTO_ERROR2_WRITING_LASTBYTE = -52
Public Const CRYPTO_BAD_LASTBYTE = -61
```

Examples :

```
Dim Test As Long
```

```
Test = cRUBYencryptFile("c:\autoexec.bat", "c:\autoexec.tb1", "Time To Win", RUBY_MODE_DESK_LOCK)
Test = cRUBYdecryptFile("c:\autoexec.tb1", "c:\autoexec.tb2", "Time To Win", RUBY_MODE_DESK_LOCK)
```

See also :

GZIPStringCompress, GZIPStringExpand

Purpose :

[GZIPStringCompress](#) compress a string into a compressed format using GZIP compression method.
[GZIPStringExpand](#) expand a compressed string into a normal format using GZIP compression method.

Declare Syntax :

For VB 4.0 (16-Bit)

```
Declare Function cGZIPStringCompress Lib "mcsec-16.dll" (Txt As String) As String
Declare Function cGZIPStringExpand Lib "mcsec-16.dll" (Txt As String) As String
```

For VB 4.0 (32-Bit)

```
Declare Function cGZIPStringCompress Lib "mcsec-32.dll" (Txt As String) As String
Declare Function cGZIPStringExpand Lib "mcsec-32.dll" (Txt As String) As String
```

Call Syntax :

```
Test$ = cGZIPStringCompress(Txt$)
Test$ = cGZIPStringExpand(Txt$)
```

Where :

Txt\$	is the original string.
Test\$	is the compressed string.

Comments :

The compression gives the better result on TEXT string.

Examples :

```
Dim Str1 As String
Dim Str2 As String
Dim Str3 As String
```

```
Str1 = "T2WIN-32 is a powerfull DLL for VB 4.0 (32-Bit) under Win95/WinNT"
```

```
Str2 = cGZIPStringCompress(Str1)
Str3 = cGZIPStringExpand(Str2)
```

```
If (Str1 = Str3) Then Debug.Print "Success !" Else Debug.Print "Error !"
```

See also :

GZIPFileCompress, GZIPFileExpand

Purpose :

[GZIPFileCompress](#) compress a file into a compressed format using GZIP compression method.

[GZIPFileExpand](#) expand a compressed file into a normal format using GZIP compression method.

Declare Syntax :

For VB 4.0 (16-Bit)

Declare Function cGZIPFileCompress Lib "mcsec-16.dll" (ByVal FileIn As String, ByVal FileOut As String) As Long

Declare Function cGZIPFileExpand Lib "mcsec-16.dll" (ByVal FileIn As String, ByVal FileOut As String) As Long

For VB 4.0 (32-Bit)

Declare Function cGZIPFileCompress Lib "mcsec-32.dll" (ByVal FileIn As String, ByVal FileOut As String) As Long

Declare Function cGZIPFileExpand Lib "mcsec-32.dll" (ByVal FileIn As String, ByVal FileOut As String) As Long

Call Syntax :

Test& = cGZIPFileCompress(FileIn\$, FileOut\$)

Test& = cGZIPFileExpand(FileIn\$, FileOut\$)

Where :

FileIn\$	is the original/compressed file.
FileOut\$	is the compressed/original file.
Test&	<0, an error has occurred.
	>=0, the length of the created file.

Comments :

The following constants are used to explain the error code :

```
Public Const CMPEXP_FILEIN_CANT_BE_NULL = -1
' occurs when the FileIn is an empty string
Public Const CMPEXP_FILEOUT_CANT_BE_NULL = -2
' occurs when the FileOut is an empty string
Public Const CMPEXP_FILEIN_AND_FILEOUT_CANT_BE_THE_SAME = -3
' occurs when the FileIn and FileOut are the same
Public Const CMPEXP_FILEIN_CANT_BE_OPENED = -4
' occurs when the FileIn can't be opened (not valid, not exist or disk error)
Public Const CMPEXP_FILEOUT_CANT_BE_CREATED = -5
' occurs when the FileOut can't be created (not valid or disk error)
Public Const CMPEXP_COMPRESS_OR_EXPAND_ERROR = -6
' occurs when compression or expansion can't be performed (disk error)
Public Const CMPEXP_CANT_GET_FILEOUT_SIZE = -7
' occurs when the length of FileOut can be read (disk error)
```

Examples :

```
Dim FileIn      As String
Dim FileOut     As String
Dim FileOut2    As String
Dim LengthIn   As Long
Dim LengthOut  As Long
```

```
FileIn = "c:\win95\system\msjt3032.dll"
```

```
FileOut = "c:\tmp\test.gzi"
```



```
FileOut = "c:\tmp\test.ugz"
```

```
LengthOut = cGZIPFileCompress(FileIn, FileOut)
```

```
LengthIn = cGZIPFileExpand(FileOut, FileOut2)
```

See also :

