Curve Encrypt

CONTENTS

WHAT IS CURVE ENCRYPT?

Curve encrypt is a program for encrypting files on a Macintosh so that they cannot be read without a pass phrase. It works on files, folders or whole volumes, and uses the IDEA encryption algorithm, which is believed to be one of the most secure available.

ADDRESS FOR SUPPORT

Please send email to kinney@bogart.colorado.edu. Bug reports, suggestions, praise, flames, whatever are all welcome. The Curve Software PGP public key is:

```
-----BEGIN PGP PUBLIC KEY BLOCK-----
Version: 2.3

mQCPAiz+bEEAAAEEAMUbtdwYC1vY+s5559ERIvC1MT+Yaw3ozheaHcUciJe7cSA
k
k9TpAQd7iKukKnQe5kK1YtvYm0JP6fmNrcO8AmG5ukvcOlyuri618sjpXncpQ1cL
5xeV80f3JtmheGMnqAzTK8OyfJ7zRh1PhAZcT/vVzf+JGuCuVcJkEfxTVMrJABEB
AAG0K0N1cnZlIFNvZnR3YXJlIDxraW5uZXlAYm9nYXJ0LmNvbG9yYWRvLmVkdT6J
AJUCBRAtBLJRIDvxOj7zTo0BAQLFA/9fmt+S3PyHcl4OpfRz0iGhtYvfq9gZW/Oq
vxWJiQBExgjtDhwq6keAO6c75D7MqJJKxIUGXOU97h92DmEn54M5SKtwVGDPkG7
8
l3WMDA90SUAdzhbXbKKKtO+rgeCBHuPftoI/PXGxSaDNspuIzoUjpIpNYrR9o6he
gIJsbMDi/YkAlQIFECz/Arz37+E6SINj8QEB43kD/R8Vfk6fhnFz+C410Nv6cdlx
3pPAnFRv1JNOWwlfgEAoBx/TEbgNjQv70M3Q3rDoU5HdG5kgBTHbnFL3JEFIwt7W
A7Dqoj0L+W06+HvJygoKQ4Gqh7qiKxHNESEivdT4VBdEi7tCGfkRMSWNGNa9Dp
+F
6iGsuFIZWx+kFoq1vZ9X
=OBJZ
-----END PGP PUBLIC KEY BLOCK-----
```

All official releases will be validated by a signature from this key.

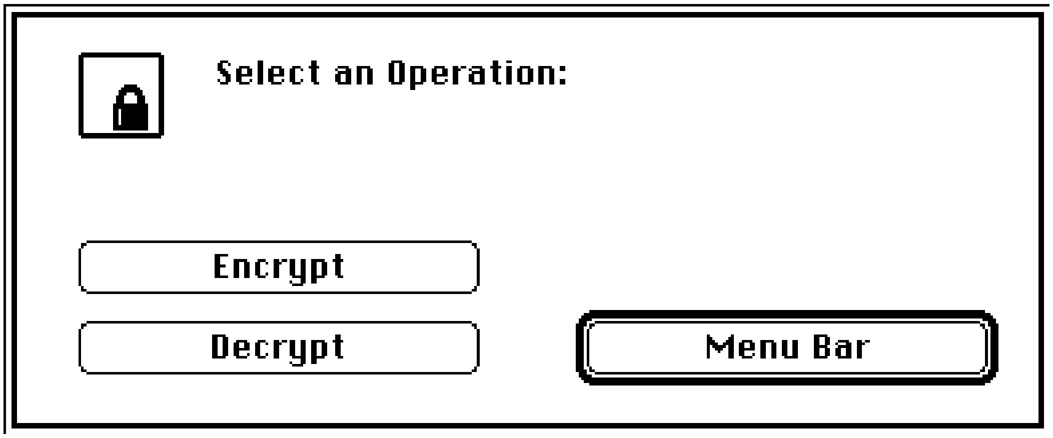
Curve Encrypt is provided free of charge. If you like the program, give it to a friend or two. If you like the source code, lift it and make something new with it. Permission is granted for distribution within the United States only.

Export of this software may violate Federal law (even though it uses a Swiss algorithm – go figure), and support will not be provided for non-U.S. addresses or anon id’s. Sorry about that.

USING CURVE ENCRYPT

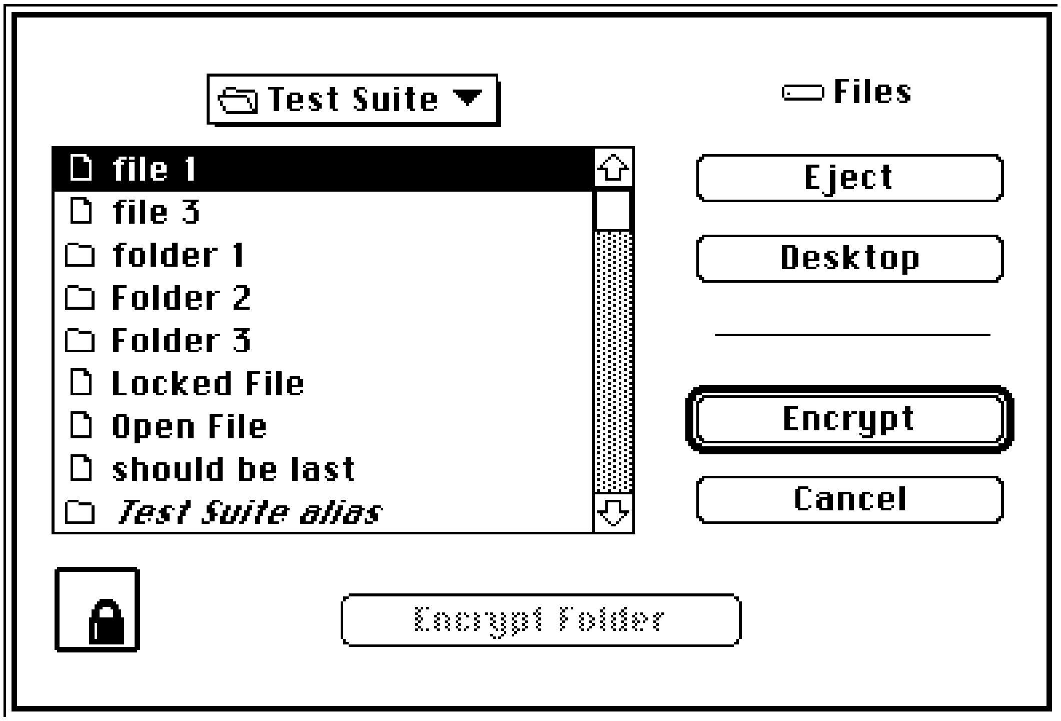
General

Curve Encrypt is simple and intuitive to use. When the application starts up, you will be provided with a dialog like this:



When you select Menu Bar, the startup dialog disappears and you can get to the menu bar. You can then select Encrypt or Decrypt from the File menu to process files or folders. Options for Curve Encrypt (discussed below) can be set by selecting Options... from the File menu.

When you choose either Encrypt or Decrypt, a file selection menu will come up:

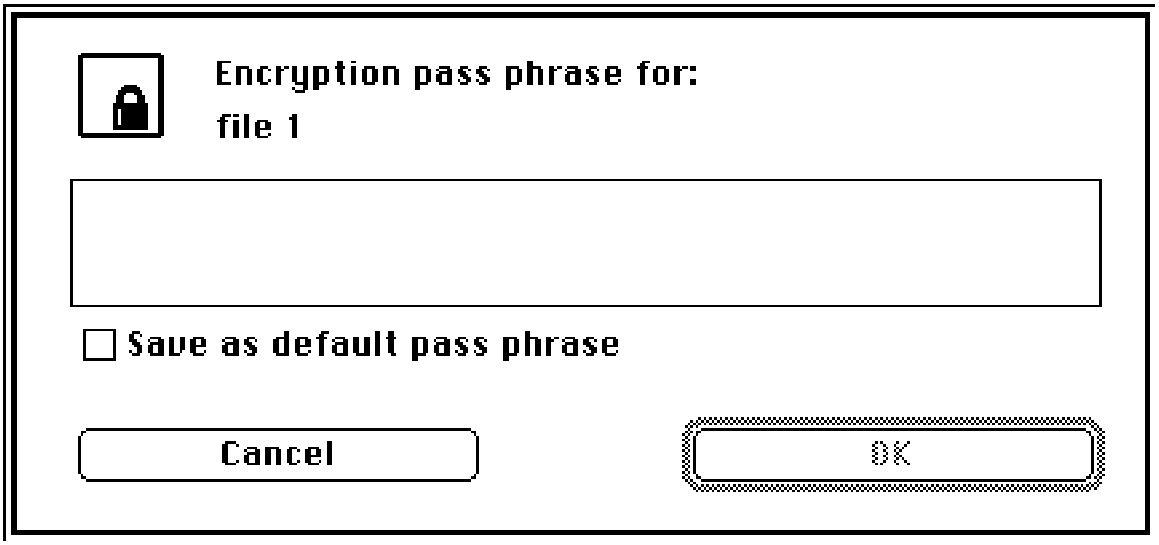


This is just like the file selection menu for any other application, except that it has an additional button titled either Encrypt Folder or Encrypt Volume depending the items you've selected in the dialog. Selecting Encrypt Folder allows you to encrypt everything in a folder with the same pass phrase in one operation. Same for Encrypt Volume.

You can also encrypt or decrypt items by dragging them on top of the Curve Encrypt icon, or decrypt by double-clicking on encrypted files. In these cases, CE will exit upon completing the requested operation.

Encrypting files

When you select a file for encryption, Curve Encrypt will ask you for a pass phrase:

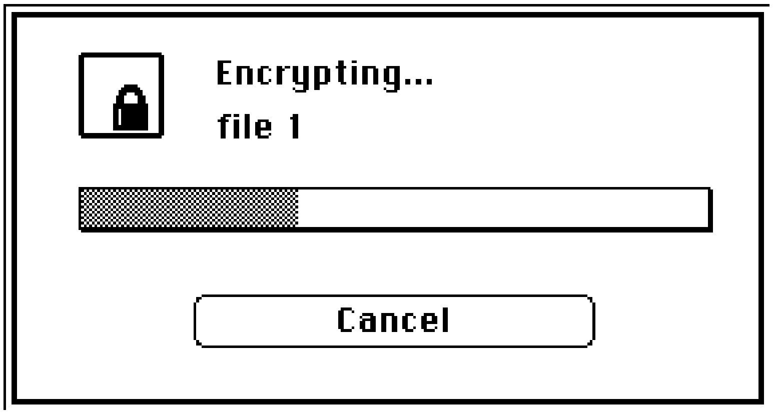


Encryption pass phrase for:
file 1

□ Save as default pass phrase

Enter a pass phrase of up to 255 characters and click OK -- note that what you type is not shown on the screen. Curve Encrypt will then ask you to re-type the pass phrase to verify that you typed what you really wanted to type.

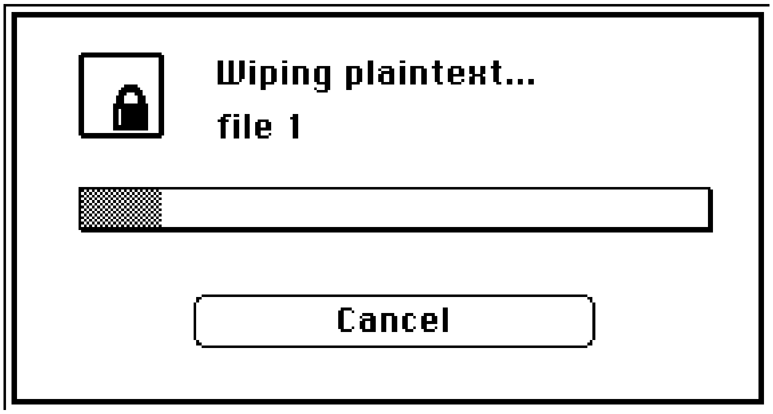
You'll then see a progress bar titled Encrypting...:



Encrypting...
file 1

You can cancel the operation by clicking Cancel, and the file will be left in its original, unencrypted state.

When the encryption is complete, you’ll see a progress bar titled Wiping Plaintext...:



Wiping plaintext...

file 1

This means that Curve Encrypt is wiping the original, unencrypted information from your disk. If you like, you can select an option to wipe the plaintext *three* times with random data. This is to protect against devices which can read data from a disk even after it’s been overwritten. See the section “Curve Encrypt Options”, below.

The Cancel button on the Wiping Plaintext... status bar has a peculiar behavior, intended to ensure security. When you select Cancel from the dialog, you will be asked to confirm, and then the program will continue wiping the file, but now with the Cancel button dimmed. This is to make sure that no plaintext is ever left on your disk. When the wipe is complete, the encryption process will be canceled.

Encrypting folders

Select Encrypt Folder or drag the folder’s icon onto Curve Encrypt. Encryption then proceeds the same way as for files. Everything contained in the folder, including other folders, is encrypted with the pass phrase you enter.

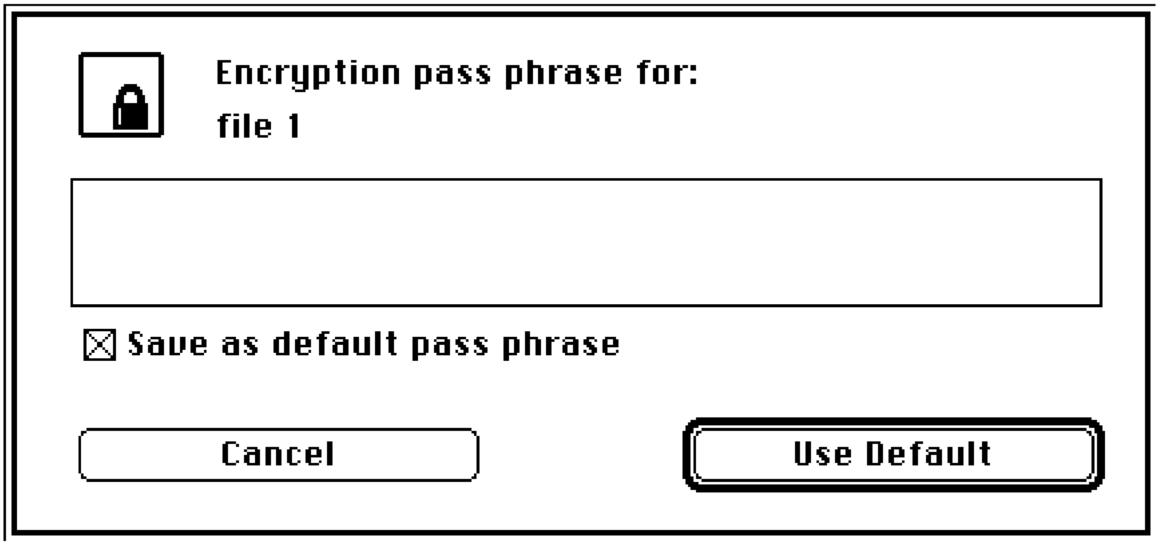
Encrypting volumes

Select Encrypt Volume or drag the disk’s icon onto Curve Encrypt, and enter a pass phrase as usual. Curve Encrypt knows not to encrypt Mac System files, so bootable disks will stay bootable. Note that (unfortunately), there must be enough free space on the volume to make a copy of the largest file on the volume, or the encryption will fail.

Encrypting multiple items

If you want to encrypt several files or folders on your disk, but don't want to have to enter some long pass phrase over and over, you can define a default pass phrase in Curve Encrypt.

On the pass phrase dialog, there's a box titled Save as default pass phrase. If you check this box when you enter a pass phrase, Curve Encrypt will remember what you entered, and the next time the pass phrase dialog comes up, you can select Use Default to use the saved pass phrase instead of entering a new one:



Encryption pass phrase for:
file 1

☒ Save as default pass phrase

You can enter a new pass phrase if you like. If the Save as default pass phrase box is checked when you enter the new pass phrase, it will become the new default, otherwise, the default will remain unchanged.

The default pass phrase is not saved to disk. When Curve Encrypt exits, the default phrase is destroyed. Note also that Curve Encrypt will exit automatically after a period you can specify in case you leave it running with a default pass phrase set.

Protected Items

Some files and folders on your system cannot be encrypted. Curve Encrypt will not encrypt invisible files or certain items in your system folder, to prevent interference with the Finder or with other applications.

Files which cannot be encrypted:
Invisible files
Finder
System
Desktop Database
Clipboard
Scrapbook

Folders which cannot be encrypted:
Apple Menu Items
Control Panels
Preferences
Extensions
PrintMonitor Documents
Trash
Startup Items
Temporary Items

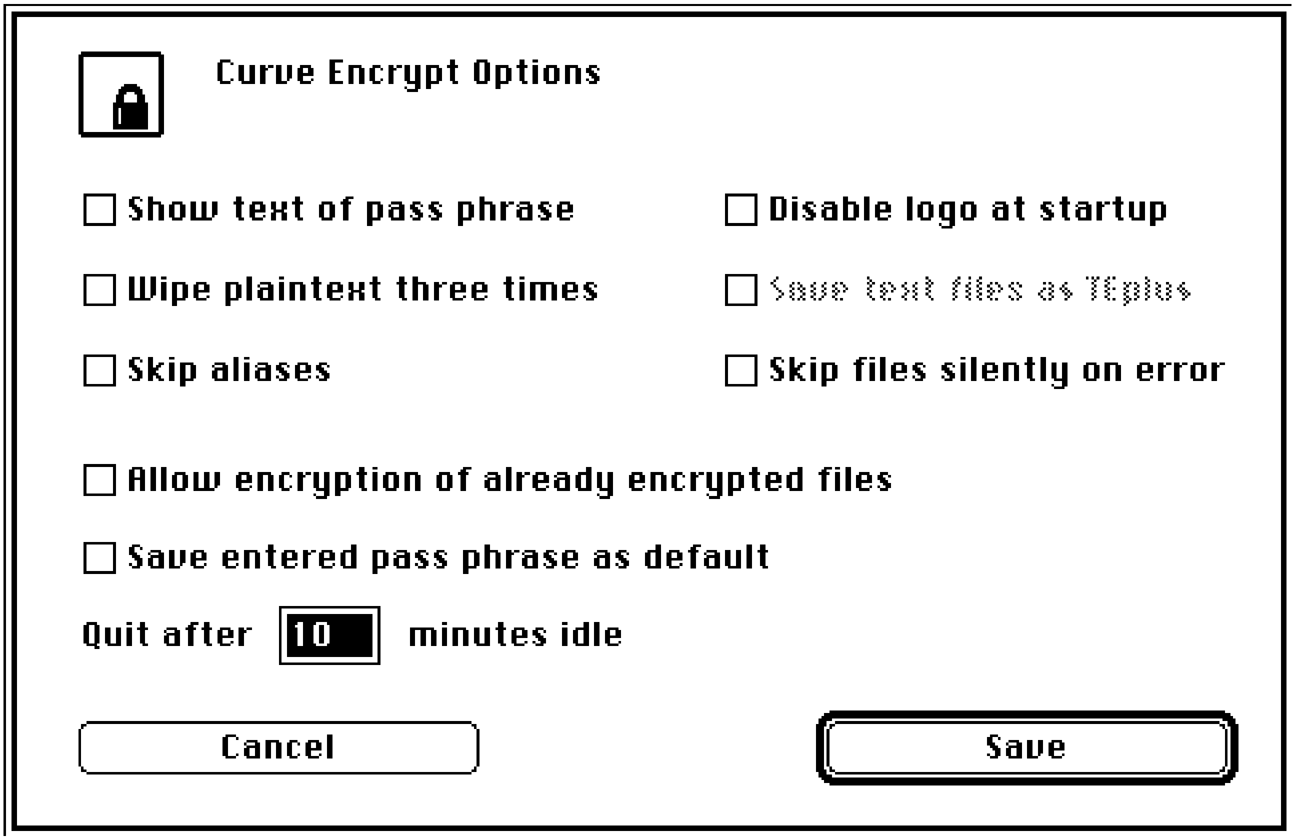
Decrypting files

Decrypting files works the same way. You're prompted for a pass phrase, and the file is decrypted (no wiping takes place). If you enter an incorrect pass phrase, Curve Encrypt will put up a dialog allowing you to skip that one file, cancel the decryption operation for all selected files, or enter a new pass phrase for the file. If you choose to enter a new pass phrase for the file, the pass phrase applies to that file only -- you cannot reset the default pass phrase without canceling the whole ball of wax and starting again.

Note that when you cancel decryption of a file, any plaintext written to your disk is always wiped.

CURVE ENCRYPT OPTIONS

Curve Encrypt has several options you can define by selecting Options... from the File menu:



Curve Encrypt Options

☐ Show text of pass phrase

☐ Disable logo at startup

☐ Wipe plaintext three times

☐ Save text files as TEplus

☐ Skip aliases

☐ Skip files silently on error

☐ Allow encryption of already encrypted files

☐ Save entered pass phrase as default

Quit after 10 minutes idle

Show text of pass phrase

When this is set, the text of the pass phrase is displayed as it is entered. When it’s not set, the text is hidden, and confirmation of the pass phrase is required on encryption.

Disable logo at startup

Keeps the “Curve Software” logo from showing when Curve Encrypt is launched. A popular option.

Wipe plaintext three times

When this is set, the plaintext file is wiped three times with random data. This is a lot slower than just wiping it once, but the it does afford additional protection against hardware which can still read data from a disk when it has been overwritten.

Save text files as TEplus

Disabled for now. This option is for compatibility with an application still in development.

Skip aliases

When this is set, aliases are skipped when you're encrypting a folder, so that only the files inside that folder are encrypted or decrypted. If it's not set, aliases are resolved and the target of each alias is encrypted or decrypted. (This option is always set when encryption of already encrypted files is allowed -- see below). Aliases can still be selected in the file selection dialog and can be dragged onto Curve Encrypt.

Allow encryption of already encrypted files

When this is checked, encrypted files can be encrypted a second time. When this option is selected, the Skip aliases option is automatically set. This is to prevent unintended double-encryption when you have a file and an alias to the file both in the same folder hierarchy.

Be careful with this option! In particular, if you have multiply encrypted files on your disk, it's probably a good idea to leave the Skip aliases option set, even if you turn off multiple encryption. This will ensure that files or folders with aliases to them will not unexpectedly be decrypted twice.

Skip files silently on error

There are several “errors” which are in response to normal conditions on your file system: locked files, files open by other applications, or an invalid pass phrase when decrypting a file. Normally, Curve Encrypt will display an error message when one of these conditions is encountered, but when Skip files silently on error is set, these files are skipped over without displaying a message. This is useful, for instance, when you have files encrypted with more than one pass phrase in a volume or folder, and want to decrypt all the files encrypted with one of the pass phrases without being repeatedly notified of invalid pass phrase errors.

Error messages which indicate other conditions, such as a full volume, are not suppressed by this option.

Save entered pass phrase as default

When this is set, the Save as default pass phrase box in the pass phrase dialog is checked automatically. The pass phrase entered is saved and can be re-used by selecting “Use Default” the next time the pass phrase dialog comes up.

Quit after ____ minutes idle

Curve Encrypt will automatically exit after the specified period if there is no activity.

HOW SECURE IS CURVE ENCRYPT?

Curve Encrypt uses the IDEA encryption algorithm, which is patented by Ashcom-Tech TG, Solothurn, Switzerland. It’s a relatively new algorithm, which means that it could be shown to be weak at some point, but it has so far withstood a great deal of expert scrutiny, and is believed to be extremely secure, more so than even the government standard DES algorithm. DES has a 56-bit key, which allows for about 7x1016 possible keys, whereas IDEA’s 128 bit key gives 3x1038 possible keys, about a thousand-billion-billion times as many. Suppose an extremely sophisticated opponent could crack DES by trying *all* the possible keys, and it took them about a day to do it: using the same technique, it would take them a billion times the age of the universe to crack IDEA. Work it out.

A good introductory article about IDEA is in Dr. Dobb's Journal, #208, December, 1993.

But no cryptographic program is 100% absolutely positively no-shit unbreakable. I've put a lot of effort into making Curve Encrypt as secure as is technically possible, but cryptography is a game of odds. Can the NSA break IDEA by some means other than an exhaustive search? Who knows? Is there a flaw in the implementation? Possibly. Public dissemination of the source code minimizes, but does not eliminate the risk of this.

Remember: the only way to make absolutely certain something can't be read is not to write it down.

All wild suspicions about the NSA aside, your worst enemy is, most likely, *you*. Having encryption software amounts to little more than a false sense of security unless you pay attention to the environment in which it's being used. Curve Encrypt is not, and is not intended to be, a complete solution to data security issues.

A few easily identifiable dangers which depend on you:

- Choose a good pass phrase! Curve Encrypt allows you to enter a pass phrase of up to 255 characters in length, caps, lower-case, spaces, funny things with umlauts over them, whatever. The pass phrase is compressed down to a 128-bit IDEA key using the MD5 cryptographic hash algorithm, so that all 255 characters in the pass phrase are significant. Make use of this. The first thing a cracker will try to do to break into a file is try a list of common names, then maybe every word in the English language, which won't take long at all. Dictionaries of common colloquial phrases are widely available. Using your phone number, your lover's middle name, or your favorite band will make you easy pickings. The commonly accepted method for composing a pass phrase is an idea called *shocking nonsense* -- a phrase which has essentially no content, but is in some way offensive or stupid enough to make it memorable. Make it something you would never actually say to anyone, so that way you never will. Something like "My Buick was eaten by giant crypto-anarchist pseudospoofer squids and no one cares. And Elvis! WHAT ABOUT ELVIS?!" would be a decent choice. People are unlikely to try that at random, or even if they know you well.

- Don't tell anyone your pass phrase, not even your dog. Don't write it down on a slip of paper. Do I have to say this?

- When you put a file in the trash, the contents of the file *are not* erased from your hard disk! Only the information which tells the operating system where to find the contents is erased. Norton Utilities' unerase program takes advantage of this fact, allowing you to recover the contents of a file that was thrown in the trash. When Curve Encrypt gets rid of a file, the contents of the file are overwritten with random data to prevent recovery, but this doesn't do anything about files just thrown in the trash. Norton Utilities, for instance, includes a "Wipe Info" program for wiping old files from your disk, so they can never be recovered.

- Text editors and other programs routinely create temporary files, which may contain whatever it is you're trying to keep secret, and be easily accessible to someone with Norton Utilities. Again, a disk wipe utility is the best solution to this kind of security leak.

- Virtual memory: When you're using Curve Encrypt, if you have virtual memory turned on on your Mac, sensitive data may be swapped out of memory and copied onto your hard disk. Curve Encrypt takes certain precautions against this -- the most sensitive data, namely your encryption keys, cannot be swapped to your hard disk, but there's no real way to plug all the leaks. Be advised that virtual memory is inherently less secure than using only RAM.

- Curve Encrypt does not hide the names of files. If you want to mask the names of the files in a folder, the simplest way to do it is to get a compression utility like StuffIt or Compact Pro and make a compressed archive of the folder, then encrypt the archive. This will also save you space, since encrypted files cannot be compressed, but compressed files are not increased (much) in size when encrypted.

The list could go on forever. The point is, the more you know about how your computer works, the safer you are. And rest assured, you haven't thought of everything.

WHAT DO I DO IF I FORGET MY PASS PHRASE?

Punt.

Your pass phrase is not written to disk with the encrypted file. So how does Curve Encrypt know you’ve entered a correct pass phrase when you want to decrypt the file? It uses a trick: when a file is encrypted, Curve Encrypt calculates a thing called a *message digest*, or *MD5 hash* of the unencrypted data in the file. The message digest is 128 bits long and depends uniquely on the contents of the file – this is encrypted along with the file data and saved to the disk. When you enter a pass phrase for decryption, the file and the message digest are both decrypted. Curve Encrypt then computes a new message digest for the decrypted file data and compares it to the original message digest – if they match, the pass phrase is good.

See the technical notes below for more information about how Curve Encrypt implements IDEA.

WHERE DO I LEARN MORE ABOUT CRYPTOGRAPHY?

On the Net, try reading sci.crypt and alt.security.pgp. For lots of political rants, try talk.politics.crypto.

The Cypherpunks mailing list contains lots of both political and technical discussion. Mail cypherpunks_request@toad.com, Subject: Help for info on the list.

Have fun.

TECHNICAL INFORMATION

This section gives a short technical description of how Curve Encrypt implements IDEA. For those of you with source, the meat of the algorithm is in the files “files.c” and “crypto.c”.

General

Curve Encrypt uses IDEA in cipher-feedback mode. The IDEA key is the MD5 hash of a 255-character pass phrase. The initial vector is derived using the IDEA cryptographic random number generator, seeded with information from the encryption key and the system clock.

Encrypted file format

Encrypted files contain both a data fork and a resource fork. The data fork contains the encrypted file, and the resource fork contains control information for decryption of the file.

Data Fork:

This is just the encrypted file, data fork and then resource fork.

Resource Fork:

Resource type ‘CINF’:

Version String (16 bytes, unencrypted):
-Version of the software used to encrypt this file.

IDEA Initial Vector (8 bytes, encrypted):
-The value of the IDEA initial vector used to encrypt the file.

Key verification block (16 bytes, encrypted):
-This is the MD5 hash of the first 4K of the plaintext file, used to verify that a decryption key is correct. See below for details.

Creator ID (4 bytes, encrypted):
-The creator ID of the plaintext file.

File Type (4 bytes, encrypted):
-The file type of the plaintext file.

Data Fork Length (4 bytes, encrypted):
-The length of the data fork of the plaintext file.

Resource Fork Length (4 bytes, encrypted):
-The length of the resource fork of the plaintext file.

Spare (64 bytes, unencrypted):
-Spare space for later use. Set to zeros.

Construction of the encryption key

The IDEA key is the MD5 hash of the pass phrase entered by the user. The pass phrase consists of up to 255 characters typed in by the user, padded to 255 bytes with zeros.

The plaintext file is encrypted using the IDEA key. The CINF resource is encrypted with the MD5 hash of the IDEA key, to help reduce the effectiveness of known-plaintext attacks on the resource data, which has a known format. The initial vector for encryption of the CINF resource is all zeros.

All buffers containing the IDEA keys, pass phrases, plaintext, and initial vectors are allocated as locked handles, and are zeroed when disposed. On machines supporting virtual memory, the memory occupied by the key is prevented from being swapped to disk by a LockMemory call, which is intended for device drivers, but works nicely here. (No modifications have been made to the IDEA or MD5 code, however, and these routines use the stack for keys, etc.)

Generation of IDEA initial vectors

Initial vectors for encryption are generated using the IDEA cryptographic random number generator, in idea.c. The generator is seeded with three values:

Key (16 bytes):
The IDEA key

Seed (8 bytes):
The seed is the xor of the first eight bytes of the IDEA key with the second eight bytes. I suppose this could be something from an MD5 of the plaintext, but this seed value seems sufficiently secure.

Timestamp (4 bytes):
The value used for the timestamp is the return from the GetTicks() function, which is the number of system ticks (1/60 second) since the last startup of the machine.

After being seeded with the above values, the random number generator is run (timestamp % seed[0]) times and the returns thrown away before initialization is complete.

Verification of Decryption Keys

The CINF resource contains a “key verification block”, which is the MD5 hash of the first 4K bytes of plaintext. When a the first block of a file is decrypted, the decrypted data is MD5 hashed to make a new key verification block, and the new verification block is compared to the old one. If they don’t match, the decryption key is flagged as bad.

File Encryption Pseudocode

```
get a pass phrase from the user
MD5 hash the pass phrase to make an IDEA key
randomly generate an IDEA initial vector

start with the data fork
while (there’s data left in the file)
        read a 4K block from the current file fork
        if (there’s space left in the key verification buffer)
                append the block to the key verification buffer
        encrypt the block
        write out the encrypted block
        if (we’re done with the data fork)
                switch to the resource fork

MD5 hash the key verification buffer to make a key verification block
Fill in the data in the CINF resource
MD5 hash the IDEA key to make a header key
set the IDEA initial vector to all zeros
encrypt the CINF resource data with the header key
save the CINF resource
```

One thing to note is that the data and resource forks of the plaintext file are encrypted as a single stream -- that is, if the last block of the data fork comes to less than 4K, the 4K encryption block is filled out with the beginning of the resource fork. The last encryption block, whether it’s from the data fork or the resource fork, is not padded to 4K before being sent to IDEA.

File Decryption Pseudocode

```
get a pass phrase from the user
MD5 hash the pass phrase to make an IDEA key

MD5 hash the IDEA key to make a header key
set the IDEA initial vector to all zeros
read the CINF resource and decrypt it with the header key

set the IDEA initial vector to the iv specified in the CINF resource
read the first 4K of the ciphertext file and decrypt it with the IDEA key
MD5 hash the decrypted data to make a key verification block

if (the key verification block doesn’t match the one in the CINF resource)
        return a bad pass phrase error

while (there’s data left to decrypt)
        read a 4K block from the ciphertext file
        decrypt the block
```

write plaintext to the data or resource fork of the plaintext file

File Wiping

All plaintext files are wiped when the encryption process is complete, with random data. The system rand generator is used for this data, as cryptographic rands are not necessary. Single or triple-wipe is specifiable in options. Ciphertext files are not wiped when a file is decrypted.

CREDITS

- Curve Encrypt ©1994, Curve Software. Permission is granted for distribution within the United States only.

- IDEA is patented by ETH and Ashcom-Tech, Solothurn, Switzerland. Swiss patent #PCT/CH91/00117.

- RSA Data Security MD5 message-digest algorithm, ©1990.

- Curve Encrypt is registered with Apple.

Cypherpunks write code!