Description of Cryptographic Architechture

This section is designed for those who know something about cryptography to get a clear picture of exactly what kind of security CryptDisk offers. It outlines the exact procedures followed in the encryption. One of the primary tenets of a secure system is that the design be public. If a system relies on its design being private for its security, that is a major weakness. The basic algorithms used here have been around for at least 2 years, and no remotely successful attacks have been discovered. They are considered extremely secure. As described above, CryptDisk source code is available to those who register for it so that third parties can insure the integrity of the algorithms.

IDEA (International Data Encryption Algorithm) was developed by Xuejia Lai and James Massey, and was finalized in 1992. Noted cryptographer Bruce Schneier called it "the most secure block algorithm available to the public at this time." IDEA uses 128 bit keys to encrypt 64 bit blocks. By contrast, the US government's DES (Data Encryption Standard) uses only 56 bit keys. To quote Bruce Schneier's book "Applied Cryptography":

Assuming that a brute-force attack is the most efficient, it would require [2 to the power 128]([10 to the power 38]) encryptions to recover the key. Design a chip that can test a billion keys per second and throw a billion of them at the problem, and it will still take [10 to the power 13] years – that's longer than the age of the universe.

Another algorithm used is a secure one way hashing algorithm called MD5 originally designed by Ron Rivest (the same guy who is the R in RSA public key cryptography). It produces a 128 bit hash of an input, and is considered one of the most secure one way hashing algorithms known to the public.

Just having good algorithms doesn't bring you very close to a secure system without a good implementation. The following paragraphs describe step by step exactly how CryptDisk uses these algorithms.

Passphrase entry for the user is limited for no particular reason to 128 bytes, and requires a minimum of 8 bytes enforced by the dialog(this is also an arbitrary limit). CryptDisk first uses the MD5 algorithm to produce a 128 bit hash of the passphrase. This hash value is then MD5 hashed together with 8 bytes of "salt", or random data which is retrieved from a random pool of data that is seeded with information from various different sources including the user mouse waving dialog. The salt is stored in the clear in the resource fork of CryptDisk files. It is different for each file. The result is then MD5 hashed an arbitrary number of times, based on the maximum number of hashes the running Macintosh can perform in half a second, with various other pieces of information. The final value is the 128 bit user key for IDEA encryption. The user key is only used to encrypt an actual session key which is generated from random data and then encrypted with the user key once. The session key is the IDEA key used to encrypt and decrypt data on the CryptDisk, and it is stored encrypted by the user key (a second time) in the resource fork of CryptDisk files. This is what allows users to change their passwords. CryptDisk need only reencrypt the stored session key with a new user key in order to change the password. The session key is encrypted the first time to insure that it is impossible to predict the random values obtained for the key. That encryption stage is never reversed. The second encryption is to actually protect the security of the key when stored in the file.

CryptDisk stores encrypted 64 bits worth of the 128 bit MD5 hashed and salted user key in order to check that a correct password has been entered. These bits are encrypted with the hashed user key itself. Once a correct password has been entered, the session key is decrypted and the file is mounted as a disk.

Encryption and decryption of the sectors on the disk is done with IDEA on sector boundaries (512 bytes). Each sector has its own IV calculated based on the data actually in the sector, and uses IDEA in CFB (cipher feedback) mode to encrypt. To generate the IV, the first 504 bytes are hashed together into a 64 bit value. This hash is XOR'd with the original 8 bytes of salt from the user key and then hashed with the block number to make sure that no block and no separate CryptDisk file ever encrypts to the same ciphertext. The final hash value is XOR'd with the last 8 bytes of the sector to become the IV for the block. The whole block is then encrypted with that IV and the write operation is completed.

Since our IV has now been encrypted, decryption must begin from after the first 8 bytes, using the first 8 as our IV. The last 504 bytes of the block can be decrypted which provides us with the original IV. This is used to decrypt the first 8 bytes which allows us to calculate the hash check value performed above and XOR that with the last 8 bytes to complete the decryption of the block.

The method described above retains the highest security by using CFB with an IV while not requiring that massive amounts of space be used to store IVs or some other method. The IV is calculated directly from the data encrypted with the IV. The IV and every bit of the following ciphertext depends on every bit in the block. A similar method was first used in the Secure File System(SFS) program available for DOS machines designed by Peter Guttman.

When the disk is unmounted, any memory data structures are cleared to zero leaving no remnant of sensitive data. The only danger is that other programs which may read data off of the CryptDisk may store it in unprotected areas. This could be done by a word processor creating temp files or by the data being swapped to an unencrypted disk by virtual memory. These are issues that are outside the scope of CryptDisk, but should be taken note of by the user. None of CryptDisk's structures will ever be swapped to disk by virtual memory because CryptDisk explicitly prevents that.