EncFS

From Wikipedia, the free encyclopedia

EncFS is a Free (GPL) FUSE-based cryptographic filesystem that transparently encrypts files, using an arbitrary directory as storage for the encrypted files.

Two directories are involved in mounting an EncFS filesystem: the source directory, and the mountpoint. Each file in the mountpoint has a specific file in the source directory that corresponds to it. The file in the mountpoint provides the unencrypted view of the one in the source directory. Filenames are encrypted in the source directory.

Files are encrypted using a volume key, which is stored encrypted in the source directory. A password is used to decrypt this key.

EncFS

Developer(s)	Valient Gough
Stable release	1.7.2 / September 5, 2010
Operating system	Linux, FreeBSD, Mac OS X
Type	filesystem, encryption
License	GPL
Website	EncFS home (http://www.arg0.net/encfs)

Contents

Advantages

EncFS sports several advantages over other disk encryption software simply because each file is stored individually as an encrypted file somewhere else in the host's directory tree.

- EncFS "volumes" do not occupy a fixed size — they grow and shrink as more files are added to or removed from the mountpoint.
- It is possible for some directories on the mountpoint to exist on different physical devices, if a filesystem is mounted over one of the subdirectories in the source directory.

- Backup utilities can back up only the files that have changed in the source directory.
- Corruption of data is more isolated. Data corruption of filedata is local to a single file and data corruption of the filesystem can be corrected with a reliable filesystem repair utility like fsck. In some whole-disk encryption systems, one or both of these attributes are not present.

Disadvantages

There are some drawbacks to using EncFS.

- EncFS volumes cannot be formatted with an arbitrary filesystem. They share the same features and restrictions as the filesystem containing the source directory.
- Fragmentation of the encrypted volume causes fragmentation of the filesystem containing the source directory.
- Anyone having access to the source directory is able to see how many files are in the encrypted filesystem, what permissions they have, their approximate size and filename length, and the last time they were accessed or modified.

Filesystem options

When creating a new EncFS volume, several different options are available to customize the filesystem to suit various needs.

Cipher

EncFS uses whatever ciphers it is able to locate in various encryption libraries on the system. Blowfish and AES are typically available.

The cipher key length can be selected for ciphers that support variable key lengths.

Block size

Each file is encrypted in blocks, and this option controls what size those blocks are. Each time a single byte is read the entire block it is contained in must be decrypted. Likewise, for each write the block must be decrypted, altered, and re-encrypted.

The default block size of 512 is sufficient for most purposes.

Filename encoding

Filenames in the source directory can be encrypted in block or stream mode. Block mode obscures the filename length somewhat, while stream mode keeps them as short as possible, which might save space on the source directory's filesystem depending on how that filesystem manages the directory tree.

Filename IV chaining

When enabled, the initialization vector for filename encryption is derived from the file's parent directories, causing two files with the same name — but in different directories — to have different encrypted filenames.

If a directory is renamed, all files and directories contained therein will need to have their encrypted filenames re-encrypted, which can be an expensive operation. This option should be disabled if heavily-populated directories will be renamed often.

Per-file IV

When enabled, each file is encrypted with a random 8-byte initialization vector, which is stored within the encrypted file in the source directory. If this option is disabled, each file is encrypted with the same initialization vector, which can make the volume key easier to break.

Enabling this option makes the filesystem more secure at the cost of an additional 8 bytes per file.

External IV chaining

Causes the file data initialization vector to be derived from the filename's initialization vector chain. The same data will be encrypted differently given a different filename or directory.

Consequently, renaming a file when this mode is enabled requires that either the file's random initialization vector be offset by the change in the filename initialization vector chain, or the data be re-encoded. The authors of EncFS have chosen the former route as it is considerably faster, especially for large files.

Block MAC headers

Stores a checksum with each encrypted block, causing corruption or modification of the encrypted files to be detected by EncFS. The checksum is 8 bytes, and optionally up to 8 additional bytes of random data can be added to each block to prevent two blocks with the same unencrypted data from having the same checksum. This option creates a large amount of CPU overhead, as each block's checksum must be calculated when data is read (to verify integrity) or written (to update the checksum).

Secondary volumes

EncFS supports a somewhat primitive form of secondary volumes, that is, a single source directory offering different files given different passwords.

If EncFS is unable to decrypt a file with the volume key, it is ignored. If EncFS is forced to ignore an invalid password entry, the volume key will decode differently, and hence files will be encrypted and decrypted with a different key. This will present two different encrypted volumes given different passwords.

However, it is possible that two filenames on two different secondary volumes will be encrypted to the same filename. In this case, any other file will be overwritten with a new file being created. Note that this refers only to the encrypted filenames, not the unencrypted filenames. This danger can be averted by creating one directory per secondary volume and storing files in the only visible directory after a secondary volume is mounted.

Also, if the password is changed, the volume key will be re-encoded with the new password. This will cause secondary filesystems to vanish, as the volume key will no longer incorrectly decode to the same key for a given secondary password. If the primary password is changed back, the secondary filesystems will become available again.

The EncFS author does not support this technique.

See also

- List of cryptographic file systems
- List of file systems
- Filesystem-level encryption
- Full disk encryption

External links

- EncFS website (http://www.arg0.net/encfs)
- HOWTO: EncFS in Ubuntu and Fedora GNU/Linux (http://www.movingtofreedom.org/2007/02/21/howto-encfs-encrypted-file-system-in-ubuntu-and-fedora-gnu-linux/)
- encfs version 1.3.2 for Mac OS X (http://encfs.darwinports.com/dports/fuse/encfs/)

Categories: Disk encryption | Free special purpose file systems | User space file systems