Функциональное описание FES

Сургученко Т.С., ФРТК, 013 гр., 4 курс

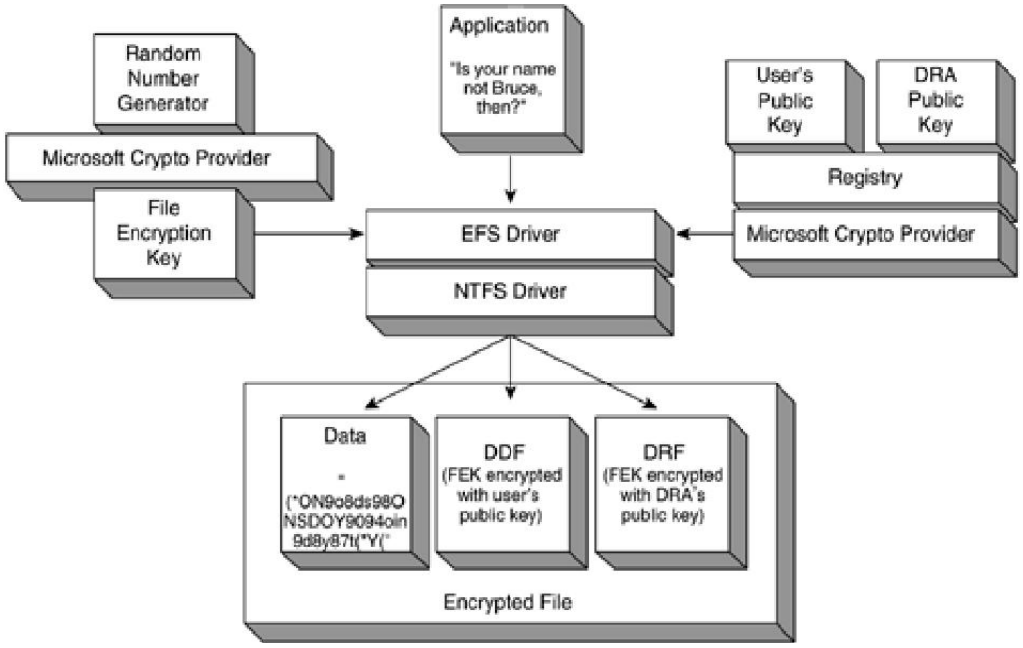
2004г

Введение

Система Шифрования Файлов (File Encryption System, FES) представляет из себя мощный встроенный механизм обеспечения безопасности данных на диске путём их шифрования. Впервые система появилась в Windows 2000, затем улучшенные версии появились Windows XP и Windows 2003. FES необходима как пользователям настольных и переносных компьютеров для защиты персональных данных, так и администраторам больших распределённых систем хранения и обработки данных для защиты корпоративной информации от атак злоумышленников.

Описание процесса шифрования файлов

FES использует множество подсистем Windows. На блочной диаграмме показаны её основные компоненты.



1. Когда пользователь выставляет флаг шифрования для файла, драйвер FES обращается к Microsoft Crypto Provider, который генерирует Ключ Шифрования Файла (File Encryption Key, FEK).

2. Microsoft Crypto Provider использует генератор случайных чисел, чтобы получить 128-битный ключ.

3. EFS использует FEK чтобы зашифровать файл. При этом шифруются только данные или именованные потоки данных. Другие атрибуты файла (имя, временные марки, атрибуты и так далее) не шифруются. Указатель на дескриптор

безопасности, показывающий,, кто имеет право доступа к файлу, так же не шифруется, хотя в файловой системе существуют дополнительные ограничения на работу с шифрованными файлами.

4. EFS сохраняет зашифрованный файл в системе NTFS, которая хранит зашифрованные данные так же, как и любые дрругие приходящие от приложения. Таким образом, EFS не заменяет NTFS, она её дополняет.

5. Затем EFS ещё раз обращается к Microsoft Crypto Provider, чтобы получить пользовательский публичный ключ EFS. Она использует этот ключ, чтобы зашифровать FEK, который затем будет сохранен в Поле Дешифрования Файла (Data Decryption Field, DDF).

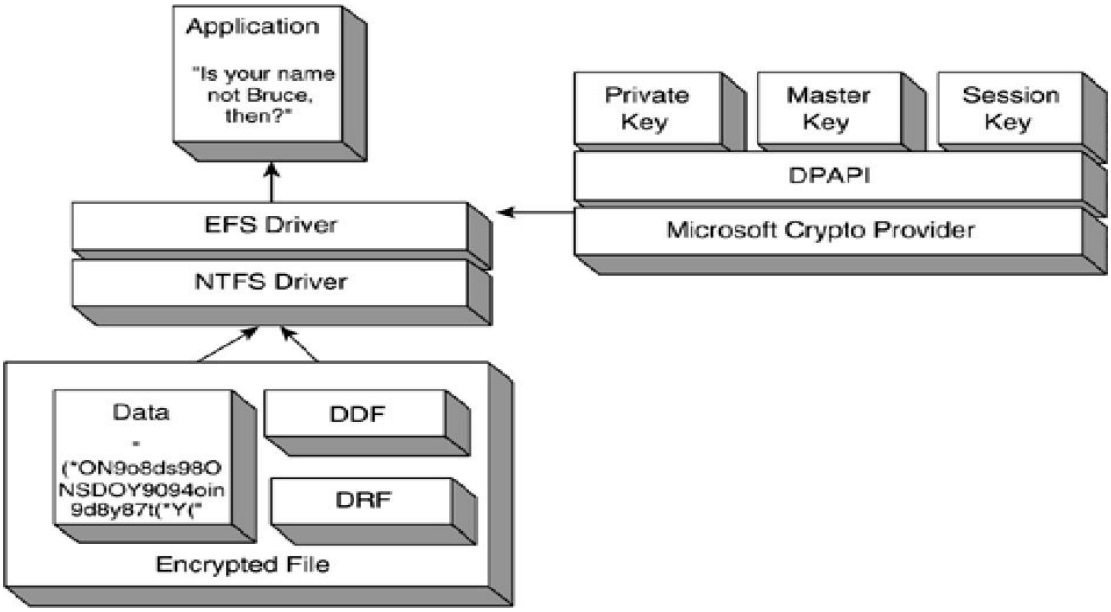
6. EFS шифрует ещё одну копию FEK с помощью публичного ключа Восстановления Файлов (File Recovery, FR), и сохраняет результат в поле восстановления данных файла (Data Recovery Field, DRF). FR принадлежит аккаунту доменного администратора, который называется агентом восстановления данных (Data Recovery Agent, DRA).

7. И, наконец, EFS сохраняет DRF и DDF в файл.

В результате получается NTFS файл с бесмыслицей в атрибуте $Data, которую можно понять, только если расшифровать её с помощью FEK. Только пользователь, зашифровавший файл, или DRA, могут расшифровать этот файл, поскольку только им известен FEK.

Описание процесса дешифрования файлов

Короткое описание тех действий, которые выполняет система, чтобы дешифровать файл, когда его открывает пользователь.



1. Пользователь открывает зашифрованный файл.

2. NTFS, видя, что у файла выставлен атрибут шифрования, перенаправляет поток данных на драйвер EFS.

3. EFS обращается к Microsoft Crypto Provider чтобы получить личный ключ пользователя.

4. Затем, EFS использует личный ключ, чтобы расшифровать FEK, находящийся в Data Decryption Field файла.

5. И, наконец, EFS использует FEK чтобы расшифровать данные в файле и передать их приложению, требовавшему их.

Защита личного ключа пользователя

Личный ключ пользователя хранится в реестре Windows. Для его защиты Microsoft Crypto Provider использует сессионный ключ, который генерируется с помощью Data Protection API (DPAPI).

DPAPI генерирует сессионный ключ при помощи секрета, вычисляемого из хеша пароля пользователя. Про то, как генерируется этот секрет, можно посмотреть на www.microsoft.com/serviceproviders/whitepapers/security.asp. Вкратце, стандартный хеш пароля пользователя хешируется снова с использованием 160-битного алгоритма SHA-1. Затем он прогоняется через 4000 итераций алгоритма PBKDF2 (подробно об этом алгоритме можно прочитать на www.rsalabs.com). После этого получается псевдо-случайное число на основе хеша пароля пользователя – так называемый мастер-ключ.

Мастер-ключ затем снова шифруется, на этот раз с использованием особой функции, называемой HMAC (Hashed-based Message Authentication Code). Полученный в результате зашифрованный мастер-ключ сохраняется в реестре.

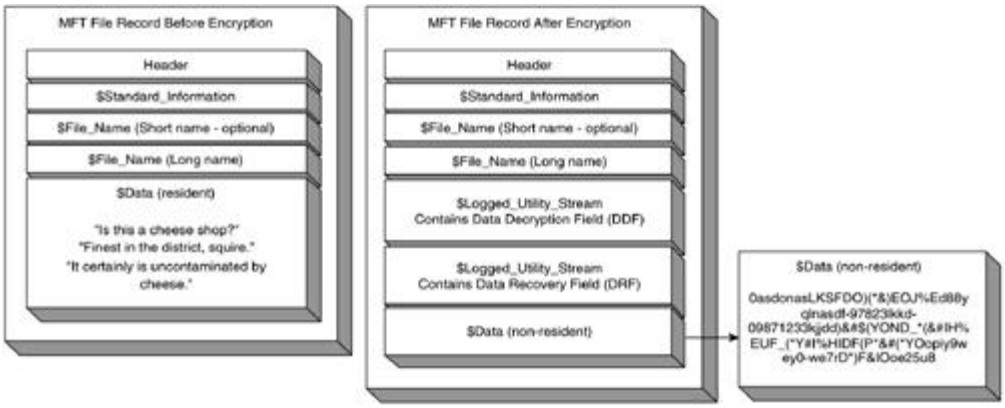
Если пользователь меняет пароль, DPAPI создаёт новый мастер-ключ, который затем используется при создании нового сессионного ключа, которым, при следующем заходе пользователя в систему будет перешифрован личный ключ.

Сертификаты

Когда пользователь первый раз шифрует файл, система получает набор ключей, выдаваемых в виде сертификатов. Если не указан Поставщик Сертификатов (Certificate Authority, CA), сертификат выдаёт Microsoft Crypto Provider. Это так называемый self-signed сертификат, означающий, что не существует цепочки поставщиков, ведущих к CA. Self-signed сертификаты используются только на отдельных машинах, таких как ноутбуки, поскольку в этом случае другие машины не могут праверить валидность поставщика.

Структура зашифрованного файла

Шифрованный файл отличается от нешифрованного только выставленным атрибутом шифрования и присутствием полей DDF и DRF. Шифрованные данные хранятся в поле $Data.



Используемые алгоритмы

В Windows 2000 для шифрования файлов использовался алгоритм DES или DESX. Используемый алгоритм выбирался доменным администратором. В Windows XP алгоритм DES убран. Зато у администратора появился выбор между DESX и 3DES. Однако, хотя алгоритм 3DES достаточно силён, тем не менее при определённых обстоятельствах он может быть взломан. Поэтому в Win XP SP1, а затем и в Win2003 алгоритмом

шифрования файлов по умолчанию является AES (Advanced Encryption Standard). Это немного неудобно, поскольку файлы, зашифрованные в системе Win2003 нельзя будет просмотреть в Win2000, однако AES даёт более мощную защиту. Кроме того, доменный администратор всегда может выбрать DESX в качестве алгоритма для шифрования по умолчанию.

Файлы, которые не могут быть зашифрованы

Практически все файлы могут быть зашифрованы, включая текстовые файлы, библиотеки, объектные файлы, временные и конфигурацаонные файлы. Существуют лишь несколько ограничений:

- **Сжатые файлы**. Шифрованные файлы невозможно сжать, и сжатые файлы невозможно зашифровать.

- **Точки монтирования.** Символические ссылки на другие файловые системы или папки не могут быть зашифрованы.

- **Системные файлы.** EFS откажется зашифровать файл с выставленным системным атрибутом. Системе может потребоваться прочитать этот файл при загрузке, ещё до того как EFS инициализируется. Невозможность сделать это может привести к невосстановимому системному сбою. Однако этим свойством можно пользоваться, выставляя системный атрибут для фалов, которые вы не хотите шифровать.

Возможные проблемы при использовании FES

- **Проверка на вирусы.** К сожалению, большинство антивирусных программ не могут получить доступ к зашифрованным фалам, поскольку они работают либо в контексте Local System Security, либо в новом контексте Local Service. Ясного решения этой проблемы нет, поскольку, теоретически, если антивирус имеет доступ к зашифрованному файлу, то, скорее всего, и злоумышленник может получить к нему доступ. Наиболее правильным решением будет использовать антивирусные сканеры в режиме реального времени, сканируя файлы в момент их открытия.

- **Сертификат DRA.** В Windows2000 файл невозможно зашифровать без сертификата DRA. Это сделано для того, чтобы пользователь не мог зашифровать файл, а затем уйти из компании, уничтожив свой приватный ключ, и тем самым оставить системного администратора без каких-либо способов расшифровать его файлы. В Windows2003 наоборот файл возможно зашифровать *без сертификата DRA*. Это нововведение имеет потенциально серьёзные последствия, поскольку при утере приватного ключа пользователь никаким образом уже не сможет расшифровать файл.

- **Алгоритмы.** Как уже упоминалось, в Windows2003 всё ещё поддерживаются алгоритмы DESX и 3DES, которые потенциально подвержены криптологическим атакам.

Список использованных источников

1. http://www.msdn.microsoft.com/library/default.asp?url=/library/en-us/dnsecure/html/WinNETSrvr-EncryptedFileSystem.asp

2. www.microsoft.com/serviceproviders/whitepapers/security.asp

3. www.ddj.com/articles/1997/9710/9710e/9710e.htm?topic=security

4. Электронная книга Inside Windows Server 2003 by William Boswell (pub. Addison-Wesley).

5. www.rsalabs.com

6. http://citeseer.ist.psu.edu/hughes01architecture.html