Эссе по курсу "Защита информации", кафедра радиотехники, Московский физико-технический институт (ГУ МФТИ), http://www.re.mipt.ru/infsec

5 Апреля 2006г. Федоткин А.С.

Защита данных в файловой системе NTFS

Физическая структура NTFS

В файловой системе NTFS каждый элемент, даже служебная информация, представляет собой файл. Самый главный файл на NTFS называется MFT, или Master File Table - общая таблица файлов. Это централизованный каталог всех остальных файлов диска (и себя самого). Он поделен на записи фиксированного размера (обычно 1 КБ), и каждая запись соответствует какому-либо файлу. MFT размещается в специальной MFT зоне (см. рис. 1) в начале диска. Первые 16 файлов носят служебный характер и недоступны операционной системе - они называются метафайлами, причем самый первый метафайл - сам MFT. Эти первые 16 элементов MFT - единственная часть диска, имеющая фиксированное положение. Для надежности, ровно посередине диска хранится их вторая копия.

В NTFS, теоретически, логические разделы жестких дисков могут быть любого размера. Максимальный размер раздела NTFS в данный момент ограничен лишь размерами жестких дисков.

В NTFS все пространство жесткого диска разделено на кластеры - блоки данных. NTFS поддерживает размеры кластеров от 512 байт до 64 КБ (неким стандартом считается кластер размером 4 КБ). Первые 12% диска отводятся под так называемую MFT зону - пространство, в которое растет метафайл MFT (см. рис 1.). Запись каких-либо данных в эту область невозможна. MFT-зона всегда держится пустой - это делается для того, чтобы служебный файл (MFT) не фрагментировался при своем росте (метафайл MFT все-таки может фрагментироваться, хотя это и нежелательно). Остальные 88% диска представляют собой обычное пространство для хранения файлов.

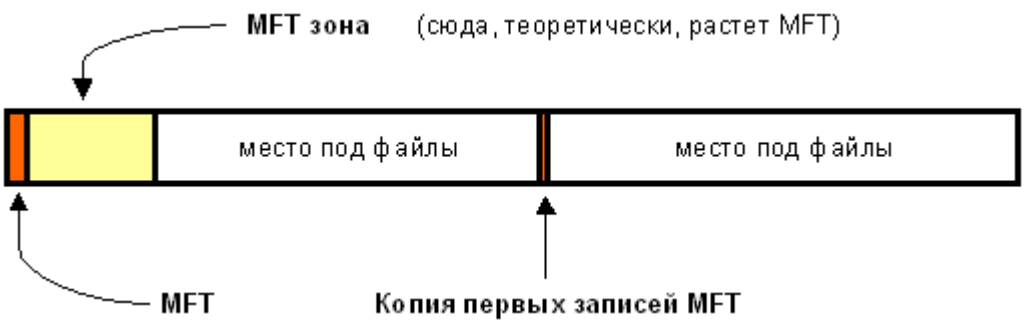

Рис.1. Структура NTFS.

Однако при недостатке свободного места на жестком диске, когда файлы уже нельзя записывать в обычное пространство, MFT-зона просто сокращается, освобождая, таким образом, место для записи файлов. При освобождении места в обычной области MFT зона снова расширяется.

NTFS может сместить (даже фрагментировать по диску) все свои служебные области (кроме первых 16 файлов), обойдя любые неисправности поверхности, поэтому, в отличие от FAT, физическое повреждение жесткого диска даже в MFT зоне не фатально для функционирования всего диска.

В MFT хранится вся информация о файле, за исключением собственно данных. Имя файла, размер, положение на диске отдельных фрагментов и т.д. Если для информации не хватает одной записи MFT, то используются несколько, причем не обязательно подряд.

Каждый файл на NTFS имеет несколько абстрактное строение - у него нет как таковых данных, а есть потоки (streams). Один из потоков и есть - данные файла. Атрибуты файлов тоже представляют собой потоки. К файлу можно "прикрепить" еще один или несколько потоков, записав в него любые данные - например, информацию об авторе и содержании файла, как это сделано в Windows 2000 и Windows XP. Эти дополнительные потоки не видны стандартными средствами: наблюдаемый размер файла - это лишь размер основного потока, который содержит традиционные данные. Можно, к примеру, иметь файл нулевой длинны, при стирании которого освободится 1 ГБ дискового пространства, - просто потому, что какая-нибудь хитрая программа прикрепила к нему дополнительный поток (альтернативные данные) гигабайтового размера.

Имя файла может содержать любые символы, включая полый набор национальных алфавитов, так как данные представлены в Unicode - в 16-битном представлении, которое дает 65535 различных символов. Максимальная длина имени файла - 255 символов.

Каталог на NTFS представляет собой специфический файл, хранящий ссылки на другие файлы и каталоги. Таким образом, создается иерархическое строение данных на жестком диске. Файл каталога поделен на блоки, каждый из которых содержит имя файла, базовые атрибуты и ссылку на элемент MFT, который в свою очередь предоставляет полную информацию об элементе каталога. Внутренняя структура каталога представляет собой бинарное дерево.

Защита данных

Сама файловая система NTFS имеет большие возможности разграничения прав доступа и защиты данных от несанкционированного доступа. Но все это можно обойти, получив физический доступ к компьютеру. Например, можно загрузиться с загрузочной дискеты и запустить общеизвестную программу NTFSdos. К тому же, получив физический

доступ к жесткому диску, можно войти на него из другой копии Windows и тогда никакие пароли не смогут защитить информацию. Единственный способ защиты от физического чтения данных - это шифрование файлов. Для этого и разработана EFS - Encrypting File System (шифрующая файловая система).

Шифрующая файловая система это тесно интегрированная с NTFS служба, располагающаяся в ядре Windows. Ее назначение: защита данных, хранящихся на диске, от несанкционированного доступа путем их шифрования. В основе EFS лежит технология шифрования с открытым ключом. Для шифрования каждого файла случайным образом генерируется ключ шифрования, зависящий от пары открытого (public) и закрытого (private) ключей пользователя. Подобный подход в значительной степени затрудняет осуществление большого набора атак, основанных на криптоанализе. При этом для шифрования файла может применяться любой симметричный алгоритм шифрования. В настоящее время в EFS используется алгоритм – DESX – специальная модификация (усиленный вариант) широко распространенного стандарта DES. DESX отличается от DES тем, что каждый бит входного открытого текста DESX логически суммируется (XOR) с 64 битами дополнительного ключа, а затем шифруется по алгоритму DES; каждый бит результата также логически суммируется (XOR) с другими 64 битами ключа. Защищенность DESX от атаки дифференциальным и линейным анализом приблизительно эквивалентна защищенности DES и потому от этих атак защищает не намного лучше. Главной причиной использования DESX является простой в вычислительном смысле способ значительного повысить стойкость DES к атакам полного поиска ключа. Создателем DESX является Рональд Ривест (Ronald Rivest).

EFS осуществляет шифрование данных, используя схему с общим ключом. Данные шифруются при помощи *ключа шифрования файла* FEK (file encryption key). FEK - это случайным образом сгенерированный ключ, длина которого составляет 56 бит. Он шифруется одним или несколькими общими ключами шифрования, предназначенных для криптозащиты ключа FEK. В этом случае создается список зашифрованных ключей FEK, что позволяет организовать доступ к файлу со стороны нескольких пользователей. Для шифрования набора FEK используется открытая пара ключей каждого пользователя.

Список зашифрованных ключей FEK хранится в специальном атрибуте EFS, который называется DDF (data decryption field - поле расшифрования данных). Информация, при помощи которой производится шифрование данных, жестко связана с этим файлом. Общие ключи выделяются из пар пользовательских ключей сертификата X.509 (сертификат авторизации).

FEK также шифруется при помощи одного или нескольких ключей восстановления (полученных из сертификатов X.509, записанных в политике восстановления зашифрованных данных для данного компьютера). Список зашифрованных ключей FEK также хранится вместе с файлом в специальной области EFS, которая называется DRF (data recovery field - поле восстановления данных). Для шифрования списка FEK в DRF используется только открытая часть каждой пары ключей. Для нормального осуществления файловых операций необходимы только открытые ключи восстановления. Агенты восстановления могут хранить свои личные ключи в безопасном месте вне системы (например, на смарт-картах).

На рисунках приведены схемы процессов шифрования, расшифрования и восстановления данных:

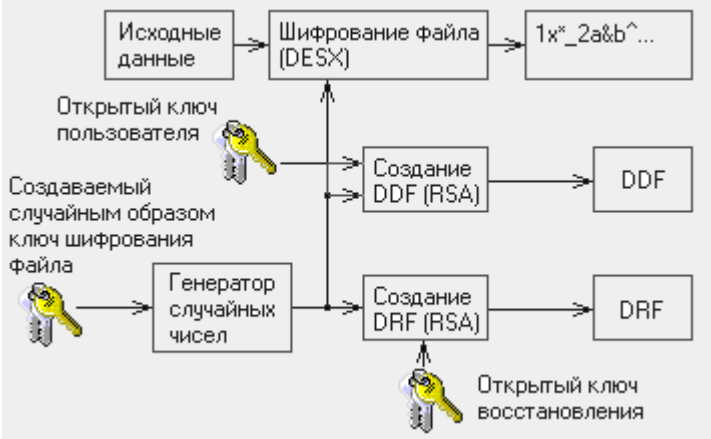

Рис.2. Шифрование данных.

Порядок шифрования:

1. Незашифрованный файл пользователя шифруется при помощи случайно сгенерированного ключа FEK.
2. Этот ключ шифруется с помощью открытой части пары ключей пользователя и помещается в поле расшифрования данных, *DDF*.
3. Ключ FEK теперь еще раз шифруется с помощью открытого ключа восстановления и помещается в поле расшифрования данных, *DRF*.

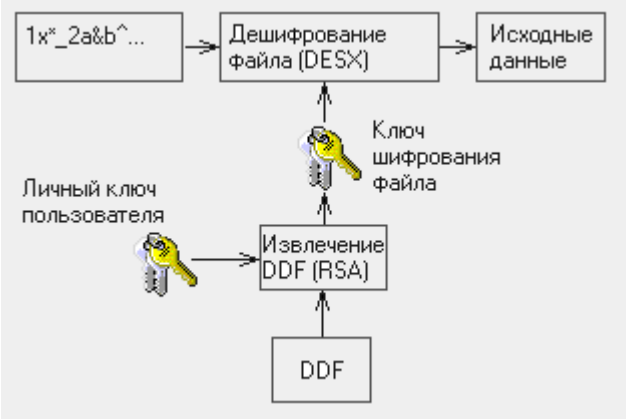

Рис.3. Расшифрование данных.

Порядок Расшифрования:

1. Из поля DDF извлекается зашифрованный ключ FEK и расшифруется с помощью закрытой части ключа пользователя.
2. Зашифрованный файл пользователя расшифруется с помощью ключа FEK, полученного на предыдущем этапе.

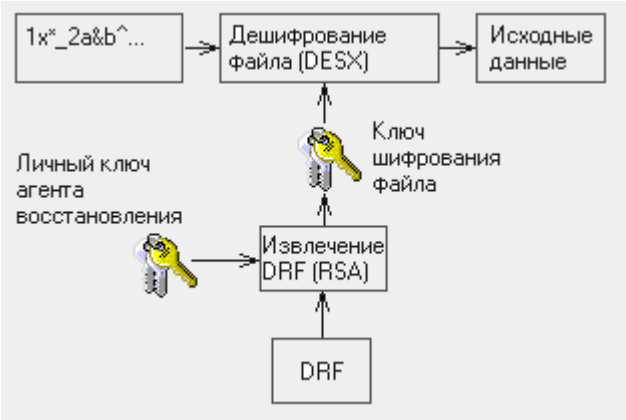

Рис.4. Восстановление данных.

Процесс восстановления:

1. Из поля DDF извлекается зашифрованный ключ FEK и расшифруется с помощью ключа восстановления.
2. Зашифрованный файл пользователя расшифруется с помощью ключа FEK, полученного на предыдущем этапе.

Реализация в Windows XP.

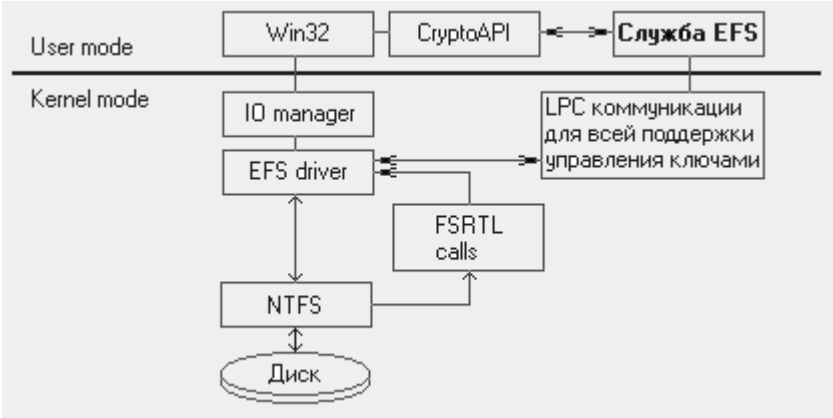

Рис.5. Архитектура EFS.

Драйвер EFS расположен на вершине NTFS. Он взаимодействует с сервисом EFS, получает ключи шифрования файлов, поля DDF, DRF и другие данные управления ключами. Драйвер передает эту информацию в FSRTL (file system runtime library, библиотека текущего выполнения файловой системы) для прозрачного выполнения различных файловых системных операций (например, открытие файла, чтение, запись, добавление данных в конец файла).

Служба EFS является частью подсистемы безопасности. Она использует существующий порт связи LPC между LSA (Local security authority, локальные средства защиты) и работающим в kernel-mode монитором безопасности для связи с драйвером EFS. В режиме пользователя служба EFS взаимодействует с программным интерфейсом CryptoAPI, предоставляя ключи шифрования файлов и обеспечивая генерацию DDF и DRF. Кроме этого, служба EFS осуществляет поддержку интерфейса Win32 API, который обеспечивает интерфейс программирования для шифрования открытых файлов, расшифрования и восстановления закрытых файлов, приема и передачи закрытых файлов без их предварительной расшифровки. Реализован в виде стандартной системной библиотеки advapi32.dll.

Список литературы:

1. Статья “Все о файловой системе WinXP”. Автор: Дмитрий Михайлов. http://www.web-support.ru/sys/win_xp_ntfs_sys.shtml.
2. “Шифрование в W2K/WinXP” (“W2k/WinXP Encrypting”). Автор: Алексей Шашков. 3D News. 29.05.2002. http://www.3dnews.ru/software/win-xp-encrypting
3. “Безопасное шифрование данных в NTFS”. Компьютерная документация от А до Я. Автор: *Евгений* Ака. Опубликовано 15 февраля 2006 года. http://www.compdoc.ru/os/windows/safe_crypt_data_in_ntfs.
4. Криптографические файловые системы, Часть первая: Дизайн и разработка. SecurityLab. 23 апреля, 2003. http://www.securitylab.ru.
5. Справочник по криптологии. Автор: *К. П. Исагулиев*. Издательство: Новое знание, 2004 г. http://www.mpgu.ru/crypto.