Государственное образовательное учреждение высшего образования

«Уральский федеральный университет имени первого Президента России Б.Н. Ельцина»

**«Защита информации»**

Семестр 4

**РЕФЕРАТ**

**Шифрующая файловая система EFS**



Преподаватель: Кибардин А.В.

Студент : Киселев В.А.

Группа : Фт-200008

Дата : 01.04.2022

Екатеринбург

2022г.

Содержание

[Введение 3](#_Toc100525877)

[Технология шифрования 4](#_Toc100525878)

[Взаимодействие с пользователем 5](#_Toc100525879)

[Восстановление данных 6](#_Toc100525880)

[Немного теории 7](#_Toc100525881)

[Процесс шифрования 8](#_Toc100525882)

[Процесс дешифрования 8](#_Toc100525883)

[Процесс восстановления 9](#_Toc100525884)

[Реализация в Windows 10](#_Toc100525885)

[Драйвер EFS 10](#_Toc100525886)

[Библиотека времени выполнения EFS (FSRTL) 10](#_Toc100525887)

[Служба EFS 11](#_Toc100525888)

[Win32 API 11](#_Toc100525889)

[Немного практики 11](#_Toc100525890)

[Выводы 14](#_Toc100525891)

[Список используемой литературы 16](#_Toc100525892)

# Введение

Шифрующая файловая система это тесно интегрированная с NTFS служба, располагающаяся в ядре Window. Ее назначение: защита данных, хранящихся на диске, от несанкционированного доступа путем их шифрования. Появление этой службы не случайно, и ожидалось давно. Дело в том, что существующие на сегодняшний день файловые системы не обеспечивают необходимую защиту данных от несанкционированного доступа.

А как же Windows NT с ее NTFS? Ведь NTFS обеспечивает разграничение доступа и защиту данных от несанкционированного доступа! Да, это правда. Но как быть в том случае, когда доступ к разделу NTFS осуществляется не с помощью средств операционной системы Windows NT, а напрямую, на физическом уровне? Ведь это сравнительно легко реализовать, например, загрузившись с дискеты и запустив специальную программу: например, весьма распространенную ntfsdos. В качестве более изощренного примера можно указать продукт NTFS98. Конечно, можно предусмотреть такую возможность, и задать пароль на запуск системы, однако практика показывает, что такая защита малоэффективна, особенно в том случае, когда за одним компьютером работают сразу несколько пользователей. А если злоумышленник может извлечь жесткий диск из компьютера, то здесь уже не помогут никакие пароли. Подключив диск к другому компьютеру, его содержимое можно будет прочитать, таким образом, злоумышленник свободно может овладеть конфиденциальной информацией, которая хранится на жестком диске.

Единственный способ защиты от физического чтения данных это шифрование файлов. Простейший случай такого шифрования — архивирование файла с паролем. Однако здесь есть ряд серьезных недостатков. Во-первых, пользователю требуется каждый раз вручную шифровать и дешифровать (то есть, в нашем случае архивировать и разархивировать) данные перед началом и после окончания работы, что уже само по себе уменьшает защищенность данных. Пользователь может забыть зашифровать (заархивировать) файл после окончания работы, или (еще более банально) просто оставить на диске копию файла. Во-вторых, пароли, придуманные пользователем, как правило, легко угадываются. В любом случае, существует достаточное количество утилит, позволяющих распаковывать архивы, защищенные паролем. Как правило, такие утилиты осуществляют подбор пароля путем перебора слов, записанных в словаре.

Система EFS была разработана с целью преодоления этих недостатков. Ниже мы рассмотрим более подробно детали технологии шифрования, взаимодействие EFS с пользователем и способы восстановления данных, познакомимся с теорией и реализацией EFS в Windows, а также рассмотрим пример шифрования каталога при помощи EFS.

# Технология шифрования

EFS использует архитектуру Windows CryptoAPI. В ее основе лежит технология шифрования с открытым ключом. Для шифрования каждого файла случайным образом генерируется ключ шифрования файла. При этом для шифрования файла может применяться любой симметричный алгоритм шифрования. В настоящее же время в EFS используется один алгоритм, это DESX, являющийся специальной модификацией широко распространенного стандарта DES.

Ключи шифрования EFS хранятся в резидентном пуле памяти (сама EFS расположена в ядре Windows 2000), что исключает несанкционированный доступ к ним через файл подкачки.

# Взаимодействие с пользователем

По умолчанию EFS сконфигурирована таким образом, что пользователь может сразу начать использовать шифрование файлов. Операция шифрования и обратная поддерживаются для файлов и каталогов. В том случае, если шифруется каталог, автоматически шифруются все файлы и подкаталоги этого каталога. Необходимо отметить, что если зашифрованный файл перемещается или переименовывается из зашифрованного каталога в незашифрованный, то он все равно остается зашифрованным. Операции шифрования/дешифрования можно выполнить двумя различными способами — используя Windows Explorer или консольную утилиту Cipher.

Для того чтобы зашифровать каталог из Windows Explorer, пользователю нужно просто выбрать один или несколько каталогов и установить флажок шифрования в окне расширенных свойств каталога. Все создаваемые позже файлы и подкаталоги в этом каталоге будут также зашифрованы. Таким образом, зашифровать файл можно, просто скопировав (или перенеся) его в «зашифрованный» каталог.

Зашифрованные файлы хранятся на диске в зашифрованном виде. При чтении файла данные автоматически расшифровываются, а при записи — автоматически шифруются. Пользователь может работать с зашифрованными файлами так же, как и с обычными файлами, то есть открывать и редактировать в текстовом редакторе Microsoft Word документы, редактировать рисунки в Adobe Photoshop или графическом редакторе Paint, и так далее.

Необходимо отметить, что ни в коем случае нельзя шифровать файлы, которые используются при запуске системы — в это время личный ключ пользователя, при помощи которого производится дешифровка, еще недоступен. Это может привести к невозможности запуска системы! В EFS предусмотрена простая защита от таких ситуаций: файлы с атрибутом «системный» не шифруются. Однако будьте внимательны: это может создать «дыру» в системе безопасности! Проверяйте, не установлен ли атрибут файла «системный» для того, чтобы убедиться, что файл действительно будет зашифрован.

Важно также помнить о том, что зашифрованные файлы не могут быть сжаты средствами Windows 2000 и наоборот. Иными словами, если каталог сжат, его содержимое не может быть зашифровано, а если содержимое каталога зашифровано, то он не может быть сжат.

В том случае, если потребуется дешифровка данных, необходимо просто снять флажки шифрования у выбранных каталогов в Windows Explorer, и файлы и подкаталоги автоматически будут дешифрованы. Следует отметить, что эта операция обычно не требуется, так как EFS обеспечивает «прозрачную» работу с зашифрованными данными для пользователя.

# Восстановление данных

EFS обеспечивает встроенную поддержку восстановления данных на тот случай, если потребуется их расшифровать, но, по каким-либо причинам, это не может быть выполнено обычным. По умолчанию, EFS автоматически сгенерирует ключ восстановления, установит сертификат доступа в учетной записи администратора и сохранит его при первом входе в систему. Таким образом, администратор становится так называемым агентом восстановления, и сможет расшифровать любой файл в системе. Разумеется, политику восстановления данных можно изменить, и назначить в качестве агента восстановления специального человека, ответственного за безопасность данных, или даже несколько таких лиц.

# Немного теории

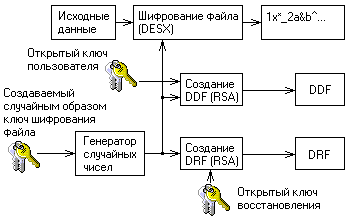
EFS осуществляет шифрование данных, используя схему с общим ключом. Данные шифруются быстрым симметричным алгоритмом при помощи ключа шифрования файла FEK (file encryption key). FEK — это случайным образом сгенерированный ключ определенной длины. Длина ключа в североамериканской версии EFS 128 бит, в международной версии EFS используется уменьшенная длина ключа 40 или 56 бит.

FEK шифруется одним или несколькими общими ключами шифрования, в результате чего получается список зашифрованных ключей FEK. Список зашифрованных ключей FEK хранится в специальном атрибуте EFS, который называется DDF (data decryption field — поле дешифрования данных). Информация, при помощи которой производится шифрование данных, жестко связана с этим файлом. Общие ключи выделяются из пар пользовательских ключей сертификата X509 с дополнительной возможностью использования «File encryption». Личные ключи из этих пар используются при дешифровке данных и FEK. Личная часть ключей хранится либо на смарт-картах, либо в другом надежном месте (например, в памяти, безопасность которой обеспечивается при помощи CryptoAPI).

FEK также шифруется при помощи одного или нескольких ключей восстановления (полученных из сертификатов X509, записанных в политике восстановления зашифрованных данных для данного компьютера, с дополнительной возможностью «File recovery»).

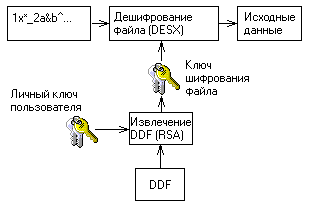
Как и в предыдущем случае, общая часть ключа используется для шифрования списка FEK. Список зашифрованных ключей FEK также хранится вместе с файлом в специальной области EFS, которая называется DRF (data recovery field — поле восстановления данных). Для шифрования списка FEK в DRF используется только общая часть каждой пары ключей. Для нормального осуществления файловых операций необходимы только общие ключи восстановления. Агенты восстановления могут хранить свои личные ключи в безопасном месте вне системы (например, на смарт-картах). На рисунке приведены схемы процессов шифрования, дешифрования и восстановления данных.

# Процесс шифрования



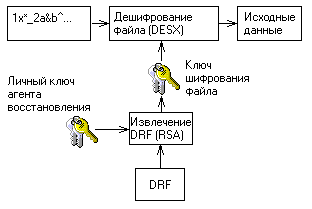
Незашифрованный файл пользователя шифруется при помощи случайно сгенерированного ключа FEK. Этот ключ записывается вместе с файлом, файл дешифруется при помощи общего ключа пользователя (записанного в DDF), а также при помощи общего ключа агента восстановления (записанного в DRF).

# Процесс дешифрования



Сначала используется личный ключ пользователя для дешифрации FEK — для этого используется зашифрованная версия FEK, которая хранится в DDF. Расшифрованный FEK используется для поблочного дешифрования файла. Если в большом файле блоки считываются не последовательно, то дешифруются только считываемые блоки. Файл при этом остается зашифрованным.

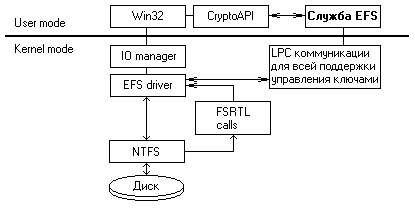
# Процесс восстановления



Этот процесс аналогичен дешифрованию с той разницей, что для дешифрования FEK используется личный ключ агента восстановления, а зашифрованная версия FEK берется из DRF.

# Реализация в Windows

На рисунке показана архитектура EFS:



EFS состоит из следующих компонентов:

## Драйвер EFS

Этот компонент расположен логически 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 времени выполнения файловой системы) для прозрачного выполнения различных файловых системных операций (например, открытие файла, чтение, запись, добавление данных в конец файла).

## Библиотека времени выполнения EFS (FSRTL)

FSRTL — это модуль внутри драйвера EFS, который осуществляет внешние вызовы NTFS для выполнения различных операций файловой системы, таких как чтение, запись, открытие зашифрованных файлов и каталогов, а также операций шифрования, дешифрования, восстановления данных при записи на диск и чтении с диска. Несмотря на то, что драйвер EFS и FSRTL реализованы в виде одного компонента, они никогда не взаимодействуют напрямую. Для обмена сообщениями между собой они используют механизм вызовов NTFS. Это гарантирует участие NTFS во всех файловых операциях. Операции, реализованные с использованием механизмов управления файлами, включают запись данных в файловые атрибуты EFS (DDF и DRF) и передачу вычисленных в EFS ключей FEK в библиотеку FSRTL, так как эти ключи должны устанавливаться в контексте открытия файла. Такой контекст открытия файла позволяет затем осуществлять незаметное шифрование и дешифрование файлов при записи и считывании файлов с диска.

## Служба EFS

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.

## Win32 API

Обеспечивает интерфейс программирования для шифрования открытых файлов, дешифрования и восстановления закрытых файлов, приема и передачи закрытых файлов без их предварительной расшифровки. Реализован в виде стандартной системной библиотеки advapi32.dll.

# Немного практики

Для того чтобы зашифровать файл или каталог, выполните следующие операции:

1. Запустите Windows Explorer, нажмите правую кнопку мыши на каталоге, выберите пункт Properties (Свойства).
2. На закладке General (Общие) нажмите кнопку Advanced.

Изображение выглядит как текст

Автоматически созданное описание

1. Отметьте галочкой пункт «Encrypt contents to secure data». Нажмите OK, затем нажмите Apply (применить) в диалоге Properties. Если Вы выбрали шифрование отдельного файла, то дополнительно появится диалоговое окно следующего вида:

Изображение выглядит как текст

Автоматически созданное описание

Система предлагает зашифровать также и каталог, в котором находится выбранный файл, так как в противном случае шифрование будет автоматически отменено при первой модификации такого файла. Всегда имейте это в виду, когда шифруете отдельные файлы!

На этом процесс шифрования данных можно считать законченным.

Чтобы расшифровать каталоги, просто снимите выделение в пункте «Encrypt contents to secure data». При этом каталоги, а также все содержащиеся в них подкаталоги и файлы будут расшифрованы.

# Выводы

* Система EFS в Windows 2000 предоставляет пользователям возможность зашифровывать каталоги NTFS, используя устойчивую, основанную на общих ключах криптографическую схему, при этом все файлы в закрытых каталогах будут зашифрованы. Шифрование отдельных файлов поддерживается, но не рекомендуется из-за непредсказуемого поведения приложений.
* Система EFS также поддерживает шифрование удаленных файлов, доступ к которым осуществляется как к совместно используемым ресурсам. Если имеют место пользовательские профили для подключения, используются ключи и сертификаты удаленных профилей. В других случаях генерируются локальные профили и используются локальные ключи.
* Система EFS предоставляет установить политику восстановления данных таким образом, что зашифрованные данные могут быть восстановлены при помощи EFS, если это потребуется.
* Политика восстановления данных встроена в общую политику безопасности Windows 2000. Контроль за соблюдением политики восстановления может быть делегирован уполномоченным на это лицам. Для каждого подразделения организации может быть сконфигурирована своя политика восстановления данных.
* Восстановление данных в EFS — закрытая операция. В процессе восстановления расшифровываются данные, но не ключ пользователя, при помощи которого эти данные были зашифрованы.
* Работа с зашифрованными файлами в EFS не требует от пользователя каких-либо специальных действий по шифрованию и дешифрованию данных. Дешифрование и шифрование происходят незаметно для пользователя в процессе считывания и записи данных на диск.
* Система EFS поддерживает резервное копирование и восстановление зашифрованных файлов без их расшифровки. Программа NtBackup поддерживает резервное копирование зашифрованных файлов.
* Система EFS встроена в операционную систему таким образом, что утечка информации через файлы подкачки невозможна, при этом гарантируется, что все создаваемые копии будут зашифрованы
* Предусмотрены многочисленные меры предосторожности для обеспечения безопасности восстановления данных, а также защита от утечки и потери данных в случае фатальных сбоев системы.

# Список используемой литературы

* <https://ru.wikipedia.org/w/index.php?title=Encrypting_File_System>
* <https://www.cryptopro.ru/products/other/efs/usage>
* <https://ru.wikipedia.org/wiki/%D0%9A%D1%80%D0%B8%D0%BF%D1%82%D0%BE%D0%BF%D1%80%D0%BE%D0%B2%D0%B0%D0%B9%D0%B4%D0%B5%D1%80>
* <https://www.kv.by/archive/index2009041105.htm>
* <https://studbooks.net/2080878/informatika/arhitektura>
* <http://adminbook.ru/index.php?men3=6/26/29>